leggi Fare Elettronica su
IE cloud
www.ie-cloud.it

fare elettronica
n.348 - Ottobre 2014

OneWireBus

come leggere più sensori senza conoscerne il RomCode

tutorial

FPGA board

per sperimentare con le FPGA

Perchè autocostruire oggi
RadioRicevitore CB
Modulazione FSK con PIC

Fare Elettronica n.348 – Ottobre 2014

Sommario

Un sistema multi OneWireBus *pag.4*

Questa applicazione introduce l'utente all'uso di più sensori termici del tipo digitale della serie DS18X20/DS1822 senza doversi preoccupare di conoscere il RomCode di ciascuna sonda.

Perche' autocostruire oggi *pag.18*

Oggi la tecnologia ha raggiunto traguardi una volta inimmaginabili. Il mercato dell'elettronica mette a disposizione dell'utente apparecchiature estremamente complesse e molto economiche. I negozi orientali forniscono qualsiasi sistema e gadget a prezzi letteralmente stracciati. E allora… perché continuare a progettare e costruire da se i circuiti elettronici?

La FPGA board *pag.36*

La scheda che ci consentirà di fare pratica cercando di capire ed imparare a maneggiare questa interessante tecnologia.

Modulazione FSK con PIC10F *pag.45*

La modulazione FSK (Frequency-Shift Keying) è una delle modulazioni più semplici che possono essere utilizzate sia in ambito hobbistico che in ambito professionale per la trasmissione di un segnale digitale tramite un collegamento analogico.

"GLUE LOGIC" *pag.51*

Scopo di questo articolo è quello di analizzare l'impiego dei microcontrollori di fascia bassa nella realizzazione di interfacce "glue logic", ampiamente utilizzate per la connessione tra diversi componenti elettronici ad elevata integrazione.

Ricevitore per la banda CB *pag.57*

Un "circuitino" da costruire con poca spesa e in poco tempo che vi darà la possibilità di ascoltare le comunicazioni in modulazione di ampiezza tra gli operatori della banda CB.

News

- Regolatore step-down µModule a tre uscite da 28VIN con dissipatore integrato fornisce 70W in 4,5cm^2

- Teledyne LeCroy presenta il nuovo software dedicato alla gestione motori "Motor Drive Power Analyzer" per gli oscilloscopi della serie HDO8000.

- Microchip presenta la prima Piattaforma di Sviluppo con 2D Multi-touch e 3D Gestures

l'elettronica è qui.

Il nuovo spazio dedicato ai progettisti elettronici e ai makers

IE cloud
www.ie-cloud.it

INWARE EDIZIONI

Il nuovo portale IEcloud mette a disposizione degli utenti numerosi ed interessanti contenuti in tema di elettronica.
Progetti, articoli e news possono essere condivisi nella community e fruiti in tempo reale da tutti i membri.

IEcloud è il portale di riferimento per tutti i professionisti, progettisti, studenti e appassionati di elettronica.

Registrati subito, è GRATIS!

- Centinaia di articoli, riviste, ebook, video, pdf sempre a tua disposizione
- Una community per condividere i propri progetti o per cercare collaborazioni
- Notizie, aggiornamenti ed eventi relativi al mondo dell'elettronica
- Un portale fruibile da qualsiasi dispositivo smartphone, tablet o PC

	FREE	SMART	MAKER	GENIUS
	GRATIS per sempre	€5.99/mese o 59,99/anno	€5.99/mese o 59,99/anno	€7.99/mese o 79,99/anno
Accesso a news ed eventi	✓	✓	✓	✓
Accesso alla community	✓	✓	✓	✓
Accesso ai progetti gratuiti	✓	✓	✓	✓
Accesso ai progetti premium	✗	✓	✓	✓
Accesso a tutte le riviste Fare Elettronica	✗	✗	✓	✓
Accesso a tutte le riviste Firmware	✗	✓	✗	✓
	Registrati	Acquista	Acquista	Acquista

Un sistema multi OneWireBus

di Alberto Trasimeni

Questa applicazione introduce l'utente all'uso di più sensori termici del tipo digitale della serie DS18X20/DS1822 senza doversi preoccupare di conoscere il RomCode di ciascuna sonda. Esistono diversi metodi per interfacciare dispositivi OneWire come DS18S20, DS18B20, o DS1822 che vanno da semplici soluzioni software, all'uso di dispositivi hardware come il DS2480B o facendo uso di controller ASIC realizzati mediante linguaggi "hardware oriented" come il VHDL.

In questo articolo prenderemo in considerazione la soluzione software,che svilupperemo mediante linguaggio di programmazione C della Mikroelektronica, nella sua versione 6.4. Per saper interpretare le diverse funzioni C presenti in questo tipo soluzione è bene disporre delle schede tecniche dei dispositivi,DS18X20 e DS1822 ottenibili dal sito web della Maxim.

Configurazione Hardware.

Lo schema a blocchi di figura 1 illustra la semplicità della configurazione hardware da realizzare quando si utilizzano più sensori di tipo OneWire.

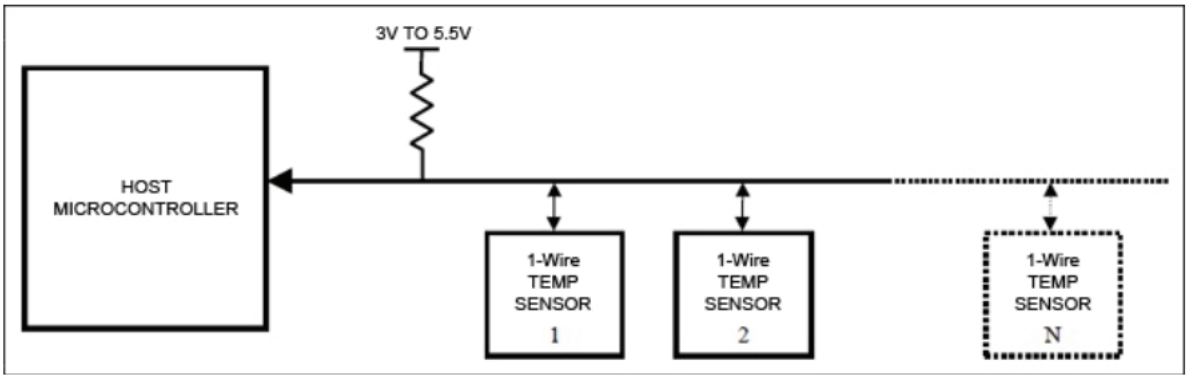


Figura 1:Interfaccia microcontrollore host

La tensione di alimentazione da usare può variare da 3.3v fino a 5.5v,il numero dei sensori che possono essere collegati al bus OneWire è quasi illimitato perché ogni dispositivo ha un identificatore del codice ROM a 64 bit e risulta univoco. In figura 1 è presente una resistenza così detta di pull-up, il cui valore deve essere di 4.7kΩ ,che consente di porre a livello logico alto il bus, nelle operazioni di lettura e scrittura delle informazioni. Tali operazioni dovranno avvenire con precisi tempi chiamati di time slot, la soluzione software consiste nell'appunto nel riprodurre mediante linguaggio di programmazione questi tempi tipici dei dispositivi. Le informazioni da leggere, il dispositivo le rende disponibili al microcontrollore attraverso il suo scratchpad, come mostra la figura 2.

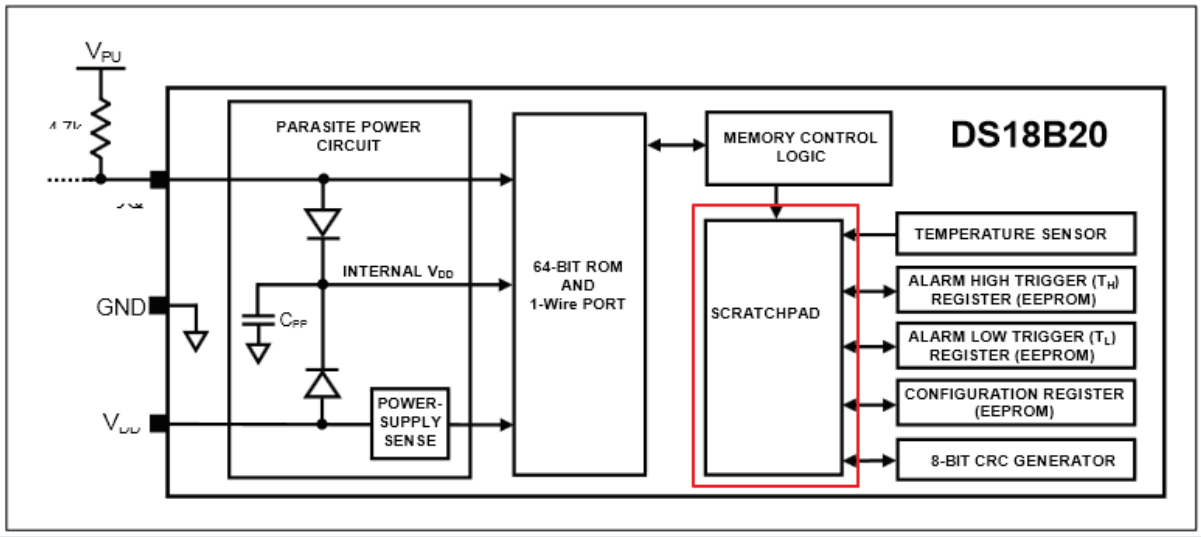


Figura 2: Diagramma a blocchi del DS18B20

Interfaccia di Timing.

La comunicazione con i dispositivi è ottenuta mediante precise temporizzazioni che consentono ai dati di essere trasmessi e ricevuti tramite il bus OneWire. Ogni comunicazione o cicli di comunicazione inizia con un impulso di reset inviato dal microcontrollore a cui segue un impulso di presenza da parte dei dispositivi locati sul bus, come mostrato in figura 3.

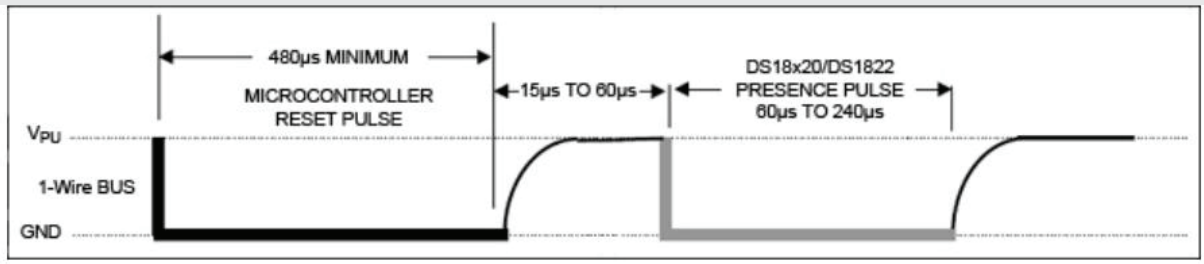


Figura 3:Impulso di reset e impulso di presenza

Perché il microcontrollore invii un impulso di reset valido, deve far passare la linea dati, dal valore logico alto a quello basso per un tempo di 480us,il dispositivo invierà la risposta con un tempo di ritardo compreso tra 15us e 60us,tale risposta consiste nel tenere la linea dati a livello logico basso per tempo compreso tra i 60us e i 240us. Vediamo ora come sono organizzati il time slot per scrivere un bit sul bus o leggere un bit dal bus.La figura 4 mette in evidenza che la loro durata è sempre di 60us indipendentemente dal tipo di operazione da compiere e dal valore del bit da scrivere o leggere. Solo che il time slot di 60us è strutturato in modo diverso il relazione al tipo di operazione e al valore del bit(solo per l'operazione di scrittura).

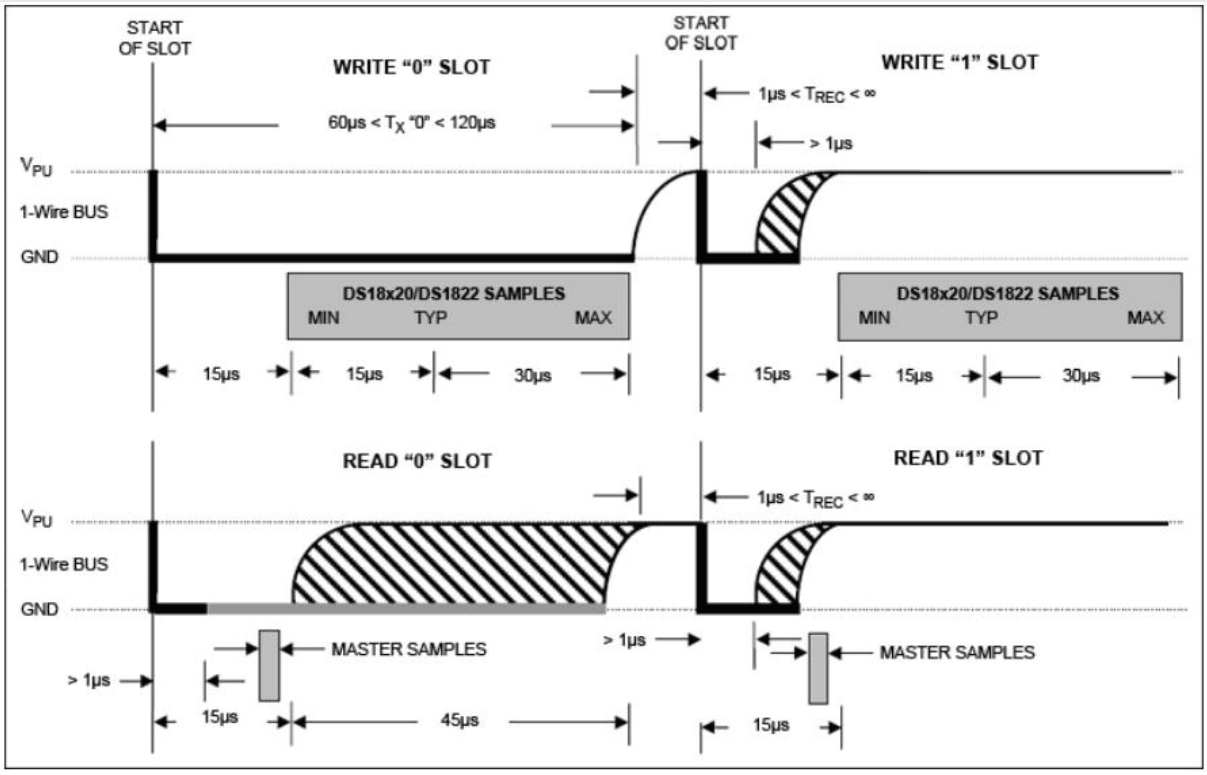


Figura 4:Scrivere e leggere il time slot

Le funzioni C che permettono il reset del bus e la scrittura di un bit, sono riportate nei listati 1 e 2 ,in cui a fianco di ogni istruzione è presente un commento relativo all'azione svolta dalla stessa.

Listato1(Reset del Bus)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
unsigned char OneWire_Reset()
{                                       // invio impulso di reset ed si aspetto la risposta
    unsigned char OneWire_Risposta;
    OneWire_Stato_Linea = 0;            // pongo la linea in uscita
    OneWire_Dato_Scrittura = 1;         // scrivo 1 sulla linea
    delay_us(10);                       // durata 10us
    OneWire_Dato_Scrittura = 0;         // scrivo 0 sulla linea(durata impulso 10us)
    delay_us(480);                      // aspetto risposta(480us)
    OneWire_Stato_Linea = 1;            // pongo la linea come ingresso
    delay_us(50);                             // aspetto 50 us prima di leggere la risposta
    OneWire_Risposta = OneWire_Dato_Lettura; // leggo la risposta(0=presente;1=assente)
    delay_us(180);                            // aspetto 180us prima di restiturire il dato
    return OneWire_Risposta;
}
```

La funzione OneWire_Reset() opera nel seguente modo: la linea dati viene posta,in un primo momento, come uscita ed al livello logico “1”,per un tempo pari a 10us.Immediatamente dopo è portata al livello logico “0”, per 480us.Attende la riposta del sensore per 50us, per poi leggere il bit e restituirlo. La restituzione avverrà dopo un tempo di 180us,considerato di sicurezza, prima di compiere altre operazioni sul bus.Il bit restituito rappresenta la risposta del dispositivo(presente=0, assente=1).

Listato 2(Scrittura del bit)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
void OneWire_ScriviBit(unsigned char OneWireBit)
{
    if (OneWireBit)
     {                                  // Scrive bit "1"
       OneWire_Stato_Linea = 0;
       OneWire_Dato_Scrittura = 0;
       delay_us(15);                    // deve essere tra 2-15 us!!!
       OneWire_Stato_Linea = 1;         // stato della linea "1" mediante pull-up
       delay_us(45);                    // completa il time slot di 60 us
     }
    else
     {                                  // scrive bit 0
       OneWire_Stato_Linea = 0;
       OneWire_Dato_Scrittura = 0;
       delay_us(60);                    // deve essere 60us!!!
       OneWire_Stato_Linea = 1;         // stato della linea "1" mediante pull-up
       delay_us(5);                     // tempo di assestament stato linea
     }
}
```

La funzione OneWire_ScriviBit(), nel tempo di time slot di 60us,sul bus scrive il bit “1” o il bit “0” in modo diverso. Se deve scrivere il bit “1” sul bus, porta la linea dati a livello logico “0” per 15us per poi portarla in ingresso(stato logico della linea “1”) per 45us. Se invece deve scrivere il bit “0” porta la linea dati a “0” per un tempo di 60us per poi portarla allo stato logico “1”.Il ritardo di 5us è stato aggiunto per sicurezza come tempo di assestamento della linea stessa. L’operazione di scrittura del bit sul bus è in perfetto accordo con i tempi di time slot forniti dalla Maxim.
Generalmente le usuali applicazioni dei microcontrollori utilizzano procedure miste, cioè fanno uso di polling ed interrupt. Nella operazione di lettura di un bit dal bus, ho preferito personalmente, disattivare gli interrupt per tutto il tempo di lettura, al fine di garantirne la corretta esecuzione della operazione. La funzione di lettura può operare anche in presenza di interrupt con l’accortezza di non superare la capacità dello

stack del microcontrollore usato. La funzione C che permette la lettura di un bit dal bus è riportata nel listato 3.

Listato 3(Lettura di un bit dal bus)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
unsigned char OneWire_LeggiBit()
{
    bit BitLetto;
    INTCON.GIEH= 0;                     // disabilito gli interrupt in lettura
    OneWire_Stato_Linea = 0;
    OneWire_Dato_Scrittura = 0;
    delay_us(2);                        // ritardo per start bit
    OneWire_Stato_Linea = 1;
    delay_us(15);                       // ritardo per la lattura
    BitLetto = OneWire_Dato_Lettura;    // leggo il dato
    delay_us(45);                       // completo il tempo di time slot 60us
    INTCON.GIEH = 1;                    // abilito gli interrupt
    return (BitLetto);
}
```

La funzione OneWire_LeggiBit() inizialmente disabilita gli interrupt porta a livello logico ”0” la linea dati per un tempo pari a 2us(ritardo start bit), per poi porla in ingresso, attendendo 15us prima di leggere il dato. Il ritardo di 45us completa il tempo di time slot in lettura. Successivamente riabilita gli interrupt e restituisce il dato letto.
Le istruzioni presenti nelle tre funzioni precedenti, ed esattamente:

1) OneWire_Stato_Linea.

2) OneWire_Dato_Scrittura.

3) OneWire_Dato Lettura.

Sono in realtà il collegamento del bus ad una porta specifica del microcontrollore come riporta il listato 4(per la serie P18).

Listato 4 (Collegamento del bus al microcontrollore)

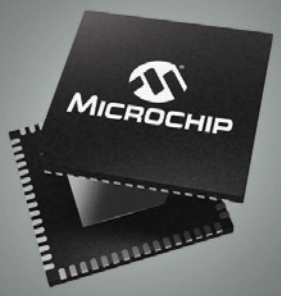
```
//--------------- OneWire sul pin6 della PORTA.A4--------------------------
sbit OneWire_Dato_Lettura    at PORTA.B4;     // lettura del pin
sbit OneWire_Dato_Scrittura  at LATA4_bit;    // scrittura sul pin
sbit OneWire_Stato_Linea     at TRISA4_bit;   // impostazione direzione linea
```

L'Intelligent Design

parte dagli Intelligent Analog PIC® Microcontroller

INTELLIGENT
ANALOG

La progettazione analogica è difficoltosa, e assorbe prezioso tempo di sviluppo. Gli intelligent PIC® MCU di Microchip integrano funzionalità analogiche come un Analog-to-Digital Converter di elevate prestazioni, Digital-to-Analog Converter, e op amp, offrendo interfacce di facile utilizzo che semplificano la progettazione analogica. Una soluzione single-chip che permette di ridurre il rumore di sistema, offre un maggiore throughput, e al contempo riduce drasticamente i tempi e costi di progettazione.



Applicazioni

- Sensori ambientali di qualità
- Apparecchiature medicali portatili
- Apparecchiature Industriali
- Conversione di potenza
- Efficienza motori
- Illuminazione
- Misurazione e controllo della potenza
- Apparecchiature di energy harvesting
- Controllo di inverter solari

microchip DIRECT
www.microchipdirect.com

MICROCHIP

microchip.com/intelligentanalog

Il nome e logo Microchip, il logo Microchip e PIC sono marchi industriali registrati di Microchip Technology Incorporated negli U.S.A. e altri Stati. Tutti gli altri marchi registrati menzionati nel presente documento appartengono ai loro titolari. © 2014 Microchip Technology Inc. TS00001743A. MEC003Ita07.14

La ricerca ROM algoritmo.

Per sfruttare appieno il concetto di bus OneWire,il microcontrollore deve essere in grado di comunicare con qualsiasi numero di dispositivi collegati alla linea dati. Per fare questo, il microcontrollore deve leggere il codice unico di identificazione ROM a 64 bit per tutti i dispositivi presenti sulla linea dati,utilizzando l'algoritmo "Ricerca ROM" come illustrato in figura 5. Una volta identificati tutti i codici ROM, il comando "Match ROM" può essere utilizzato per comunicare con qualsiasi dispositivo specifico presente su tale linea.

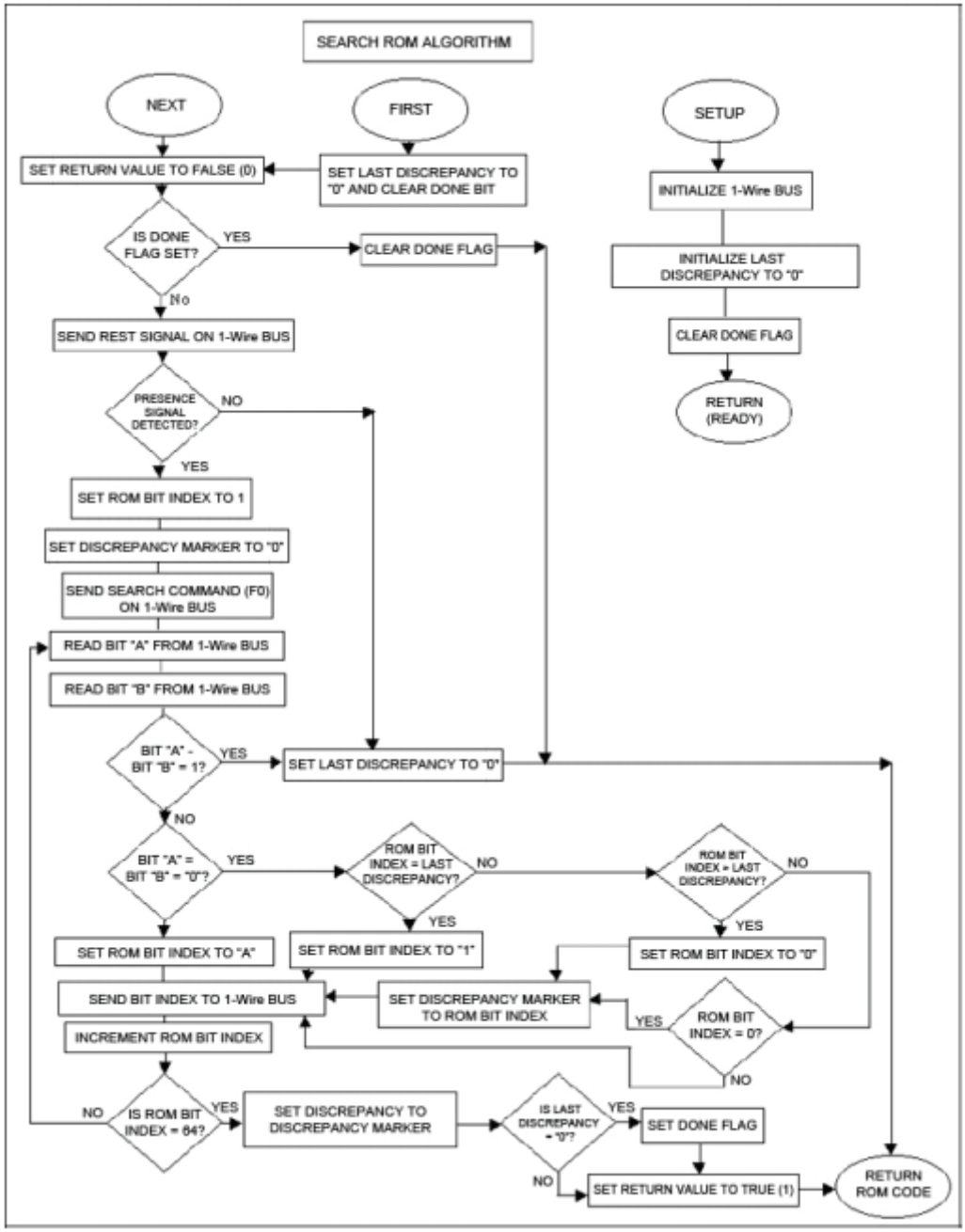


Figura 5:Algoritmo ricerca codice ROM

In effetti per leggere i ROM code di tutti i dispositivi è necessario procedere nel seguente modo:

1) Si deve leggere inizialmente il ROM code del primo dispositivo.

2) Si leggono i ROM code dei successivi dispositivi.

Per fare questo utilizzando il linguaggio di programmazione C è necessario creare due funzioni distinte, la prima legge il ROM code del primo dispositivo presente sulla linea dati(bus),la seconda leggerà il ROM code di tutti i successivi dispositivi. Riporto nel listato 5 il corpo della funzione per leggere il Rom code del primo dispositivo.

Listato 5 (Lettura ROM code primo dispositivo)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
unsigned char OneWire_RomCodePrimaDevice()
{
    LastDiscrepancy = 0;
    DoneFlag = 0;
    return (OneWire_RomcodeSuccessivaDevice());
}
```

Le variabili LastDiscrepancy e DoneFlag devono risultare essere globali e quindi essere dichiarate prima della funzione main(). Per quanto riguarda la seconda funzione quella che permette la lettura dei ROM code successivi,in effetti è una funzione abbastanza complessa, che è stata derivata e successivamente modificata in modo personale,per adattarla al linguaggio C della Mikroelektronica,da un esempio gentilmente fornito dalla Maxim.Data la sua lunghezza non mi è possibile riportanrne il listato,ma è una implementazione dell'algoritmo di figura 5.Comunque la libreria One_Wire_LIB.c completa di tutte le funzioni sarà messa a disposizione dell'editore e quindi scaricabile direttamente dal suo sito.Per completare il progetto software del multi OnwireBus,visto che i dati dai dispositivi vengono letti e scritti con la lunghezza di un byte, riporto nei listati 6 e 7 il corpo di tali funzioni.

Listato (Scrittura di un Byte)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
void OneWire_ScriviByte(unsigned char WriteByte)
{
  unsigned char i;
 for (i = 0; i < 8; i++)
  {
   // scrivo dal LSB al MSB
   OneWire_ScriviBit((WriteByte >> i) & 0x01);
  }
  delay_us(105);
}
```

Come si nota questa funzione chiama la funzione che permette la scrittura di un bit,inoltre la particolarità della trasmissione del byte al dispositivo,è che i diversi bit vanno inviati partendo da quello meno significativo,con un tempo di ritardo tra scritture successive di 105us.La funzione che legge il byte dal dispositivo (quindi i diversi byte dello scratchpad) è riportata nel listato 7.

Listato 7(Lettura di un Byte)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
unsigned char OneWire_LeggiByte()
{
   unsigned char i;
   unsigned char ByteLetto = 0;
   for (i = 0; i < 8; i++)
    {
      if(OneWire_LeggiBit())
       {
         ByteLetto|=(1 << i);
       }
       delay_us(120);
    }
   return(ByteLetto);
}
```

Durante la fase di lettura i diversi bit del byte il dispositivo li invia in modo inverso a quella di scrittura, cioè dal più significativo al meno significativo. Infine La funzione che permette la lettura di tutti i ROM code, dei dispositivi presenti sul bus e che frutta a sua volta le due funzioni di basso livello, OneWire_RomCodePrimaDevice() e OneWire_RomCodeSuccessivaDevice(),ed inoltre restituisce il numero dei dispositivi presenti sul bus, è riportata nel listato 8.

Listato 8(Lettura di tutti I Rom code)

```
unsigned char OneWire_TuttiRomCode()
{
  unsigned char cont=0;
   if(OneWire_Reset()==0)
    {
      // legge ROM primo dispositivo
      while(OneWire_RomCodePrimaDevice()==0);
      cont+=1;
      for(i = 0; i < 8; i++)
        {
          SONDA[0][i]=RomCode[i];      // memorizza in SONDA la ROM
        }
       j=1;
       while(OneWire_RomcodeSuccessivaDevice())
        {
          // legge ROM dei successivi dispositivi
          for(i = 0; i < 8; i++)
           {
             Sonda[j][i]=RomCode[i];  // memorizza in SONDA la ROM
           }
           j++;
           cont++;
        }
    }
  return cont;
}
```

Si nota che i codici dei dispositivi vengono memorizzati in una matrice chiamata SONDA, la dimensione di questa matrice dipenderà dal numero dei dispositivi utilizzati, ed in modo particolare il numero delle righe dovrà risultare essere uguale al numero dei dispositivi. Per quanto riguarda le colonne, siccome il codice ROM è costituito da 64 bit, queste dovranno essere otto. Si nota osservando le due while,che entrambe le funzioni di basso livello restituiscono valori,ed in modo particolare la funzione che legge il codice ROM del primo dispositivo deve restituire “1” se la lettura è corretta, così come la funzione che legge i codici dei dispositivi successivi per poter continuare la lettura. La variabile di stack “cont”,conterrà alla fine della funzione in numero dei dispositivi presenti sul bus. Per poter leggere o scrivere uno specifico dispositivo sarà necessario inviargli prima, di poter svolgere qualsiasi operazione, il suo ROM code. Questo compito verrà svolto dalla funzione OneWire_InviaRomCode() il cui corpo è riportato nel listato 9.

Listato 9(Invia Rom code al dispositivo)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
void OneWire_InviaRomCode(unsigned char *RomCode)
{
  unsigned char i;
  OneWire_Reset();
  OneWire_ScriviByte(OW_MATCH_ROM);
  for(i,i=0;i<8;i++)
   {
     OneWire_ScriviByte(RomCode[i]);
   }
}
```

Questa funzione utilizza la variabile “unsigned char *Romcode” variabile di tipo simbolico, in effetti al momento della chiamata della funzione si dovrà passare la variabile SONDA[x][8],in cui x rappresenta il numero del dispositivo con cui si vuole dialogare. Dopo aver strutturato gli strumenti software per la gestione del nostro multi OneWire bus, sarà necessario fare esempio di funzione per la lettura di un generico sensore termico del tipo DS18B20.Il corpo della funzione in questione è riportato nel listato 10.

Listato 10(Lettura Sensore termico)

```
void Legge_Sensore_B20(unsigned char indice,
                       unsigned int  *temperatura_var,
                       unsigned int  SONDA_Vect)
{
    OneWire_InviaRomCode(SONDA[indice]);
    OneWire_ScriviByte(0x44);              // comando start conversion
    Delay_ms(250);                         // attendo 750 ms   250
    OneWire_InviaRomCode(SONDA[indice]);
    OneWire_ScriviByte(0xBE);              // comando per leggere lo scratchpad
    Delay_us(250);                         //************************* 250
    *temperatura_var=OneWire_LeggiByte();                        // leggo la parte bassa
    *temperatura_var=(OneWire_LeggiByte()<<8)+*temperatura_var; // leggo la parte alta
}
```

Dal listato si nota che la prima operazione da effettuare è quella di inviare al dispositivo il suo ROM code, questo al fine di poter scrivere il comando che fornisce lo start conversion al convertitore interno, ed attendere 250ms.Rinviare il ROM code per scrivere il comando lettura dello scratchpad ed attendere 250ms prima di poter leggere il dato temperatura(per la prima lettura della temperatura è necessario attendere 750ms).Il numero dei bit del convertitore interno per il DS18B20 è ha

dodici, quindi e necessario utilizzare una variabile a 16 bit. La temperatura fornita dal sensore deve essere letta in due volte, cioè prima la parte bassa(dei 12 bit) e poi la parte alta. Nella variabile temperatura così letta è presente sia la parte intera che quella frazionaria. Non discuterò in questa sede il trattamento da effettuare per ottenere la parte intera e quella frazionaria dai 12 bit letti. Riporterò invece la funzione che permette di leggere il sensore, la prima volta attendendo i 750ms canonici, riportati nel foglio tecnico del sensore, per poi leggerlo con i 250ms di attesa, utilizzati nella funzione di lettura del sensore da me proposta. Posso garantire che attendendo solo(per le successive letture di temperatura) i 250ms la lettura della temperatura è sempre corretta. La funzione di Start_Up_sensore(),così come l'ho denominata è riportata nel listato 11.

Listato 11(Start_Up_Sensore)

```
/*****************************************************
 * Nome Funzione:                                    *
 *   <f_nome>                                        *
 * Descrizione:                                      *
 *   <descrizione>                                   *
 * Parametri Input:                                  *
 *   <lista_parametri>                               *
 * Parametri Restituiti:                             *
 *   <lista_parametri>                               *
 * NOTE:                                             *
 *   <note>                                          *
 *****************************************************/
void Start_Up_Sensore(unsigned char indice,
                      unsigned char *flag_lettura_var,
                      unsigned char SONDA_vect)
{
  if(*flag_lettura_var==0)                // fase di start up del sensore
   {
     OneWire_InviaRomCode(SONDA[indice]); // indirizzo la sonda
     OneWire_ScriviByte(0x44);            // comando start conversion
     Delay_ms(750);                       // attendo 750 ms
     *flag_lettura_var=1;                 // eseguita prima lettura
   }
}
```

Si vede che il problema può risolto facilmente facendo uso di un semplice flag.Il progetto multi OneWire bus da me proposto, è stato prima simulato mediante il programma Proteus 8.1 e successivamente realizzato facendo uso della scheda di sviluppo EasyPIC7 della Mikroelektronica.Le immagini 1 e 2 mostrano la simulazione ed il sistema reale.

Immagine 1(Simulazione con Proteus)

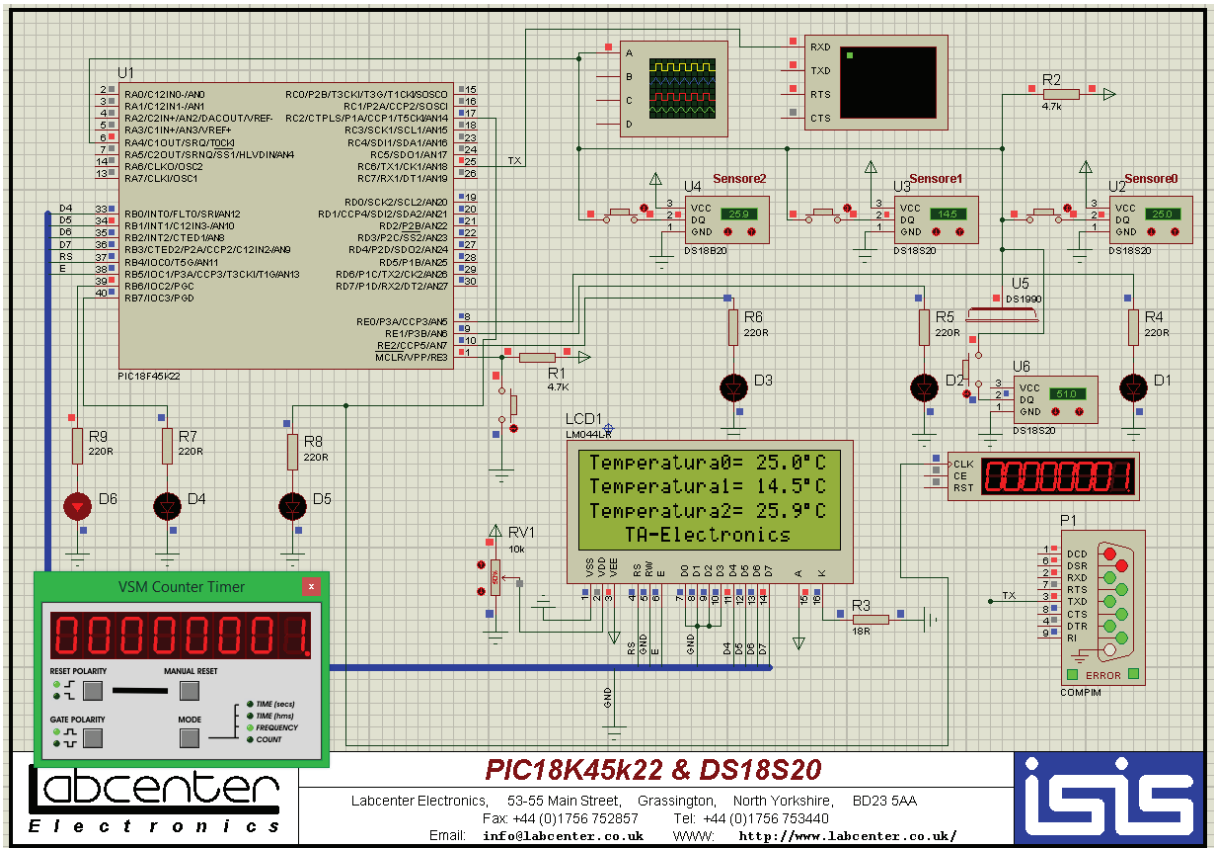
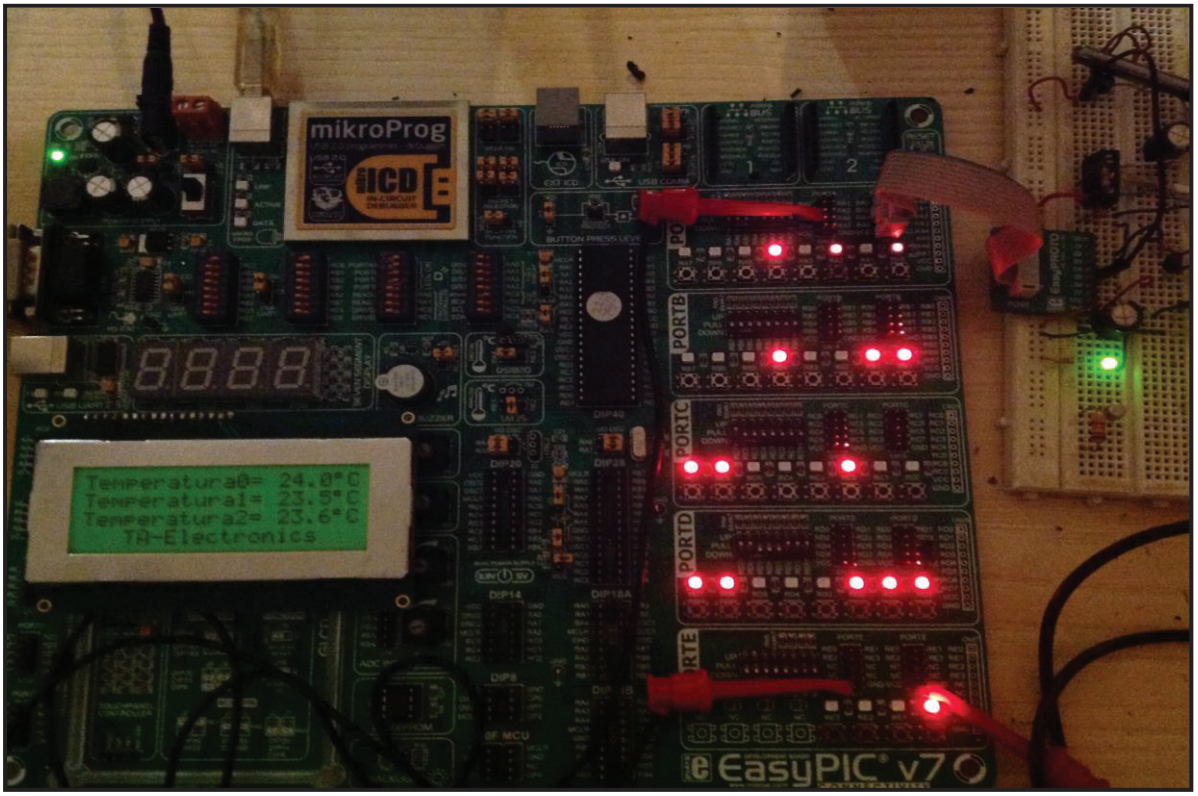


Immagine 2(Realizzazione con EasyPic7)



Conclusioni:

Il sistema da me presentato è molto utile nell'ipotesi che si abbia la necessità di monitorare più punti di temperatura con un solo microcontrollore. L'uso di dispositivi di tipo digitale permette di eliminare quasi completamente l'effetto nocivo dei rumori dovuti ad alimentazione sporca, che nel caso di sensori analogici,porterebbero inevitabilmente a spiacevoli errori di misura. L'unica precauzione da prendere in questo caso è quella di filtrare l'alimentazione dei sensori. Ricordo inoltre che i sensori DS18X20 della Maxim possono funzionare anche in modo parassita, cioè utilizzando i segnali del bus per fornire energia al dispositivo. La misura della temperatura con il DS18S20 può essere fatta con una precisione di+/- 0.5 gradi Celsius; mentre per il DS18B20 può essere facilmente misurata al decimo di grado.

Riferimenti Web:

http://www.maximintegrated.com/en/app-notes/index.mvp/id/162
http://datasheets.maximintegrated.com/en/ds/DS18S20.pdf
http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf
http://www.labcenter.com/index.cfm
http://www.mikroe.com/mikroc/pic/
http://www.mikroe.com/easypic/

PERCHE' AUTOCOSTRUIRE OGGI

di Giovanni Di Maria

Oggi la tecnologia ha raggiunto traguardi una volta inimmaginabili. Il mercato dell'elettronica mette a disposizione dell'utente apparecchiature estremamente complesse e molto economiche. I negozi orientali forniscono qualsiasi sistema e gadget a prezzi letteralmente stracciati. E allora… perché continuare a progettare e costruire da se i circuiti elettronici?

Figura 1: Un appassionato all'opera

L'hobbista elettronico da parte di una categoria che pian piano si va estinguendo. Si tratta di una persona, appassionata di elettronica (a volte senza nemmeno conoscerla) che trova sulle riviste specializzate gli schemi di cui ha bisogno e li realizza senza problemi. I più bravi riescono addirittura a disegnarli e progettarli da sé. Purtroppo la tecnologia di oggi tende a distruggere questa figura poiché mette a disposizione qualsiasi genere di sistema già pronto all'uso e a costi estremamente bassi. Perché allora si dovrebbe continuare a costruire in casa i circuiti, spendendo molto tempo e denaro col dubbio che qualche cosa potrebbe poi andare storto?

In passato

Circa 30 anni fa l'elettronica, nel campo hobbistico, era molto diffusa. Le riviste del settore erano colme di kit e articoli tecnici che insegnavano a costruire, a costi molto bassi, progetti interessanti, che divertivano molto sia chi realizzava il progetto sia chi ne assisteva al funzionamento. In quei tempi, ma ancora aggi ma in maniera veramente ridotta, si trovavano in edicola svariate riviste di elettronica e di progettazione. Esse mettevano a disposizione dell'utente migliaia di schemi pratici, di qualsiasi genere: dai gadget divertenti ai sistemi per la casa, dalle utilità per l'automobile ai circuiti radio ricevitori. Ogni progetto era corredato di schema elettrico,

foto, descrizione e illustrazione, in scala naturale, del PCB che l'utente poteva fotocopiare e riprodurre. Inoltre le pubblicazioni non si limitavano a fornire semplicemente il disegno del circuito e dello stampato, ma approfondivano gli argomenti regalando un vero e proprio corso di elettronica, con tanto di teoria e di pratica.La realizzazione del circuito elettrico costituiva un fatto molto emozionante. S'iniziava con l'acquisto dei componenti elettronici, molti dei quali irreperibili (Internet non era così diffuso) per poi passare alla realizzazione del circuito stampato in casa con diverse metodologie. Poi si procedeva alla saldatura e infine al collaudo finale. Spesso i risultati finali erano deludenti, il prototipo non funzionava subito. Si scopriva, dopo un attento controllo, che il difetto risiedeva in una saldatura fredda. A volte, invece, il difetto era colpa di un errore di stampa da parte della rivista che, puntualmente, riportava la famosa "errata corrige" che risolveva, di fatto, il problema.
Perché questa categoria di appassionati si è ridotta così drasticamente? Perché non ci si costruisce più alcuno schema elettrico come una volta? Perché il numero di riviste pratiche di sperimentazione si è ridotto all'osso, contribuendo in parte all'estinzione della specie in oggetto? Vediamo di trovare assieme le motivazioni, nel seguito dell'articolo.

Figura 2: Circuito elettrico Manhattan Style

Fattori che stanno nuocendo all'elettronica hobbistica

Di seguito si esamino alcune motivazioni che spiegano il motivo per cui l'hobby dell'elettronica sta subendo un forte calo.

La vita frenetica

Un grosso ostacolo che frena la voglia di auto-costruire è rappresentato dalla vita frenetica dei giorni nostri. La fretta, la velocità, l'uso di internet, di tablet e di cellulari mettono in condizione l'uomo a perdere quel gustoso momento di relax che ognuno poteva concedersi. Non si trova più il tempo di chiudersi in casa, col proprio saldatore, e passare un paio d'ore ad assemblare componenti. Si preferisce giocare ad un bel videogioco o sfiorare nervosamente lo schermo di un tablet.

La nuova tecnologia è difficile e complicata

L'elettronica dei giorni d'oggi non è come quella di trent'anni fa. Prima i sistemi e i circuiti elettronici (anche quelli industriali, commerciali e professionali) erano costituiti e costruiti con componenti elettronici discreti. Chiunque poteva metterci le mani per una riparazione o una sostituzione. Il disegno e la progettazione erano un fatto semplice e anche un singolo componente era abbastanza semplice da trovare. Le dimensioni della realizzazione non erano estremamente ridotte e l'intero circuito stampato aveva misure abbastanza agevoli per lavorarvi all'interno. Anche i circuiti integrati non erano molto diffusi e i pochi esemplari utilizzati disponevano di una piedinatura larga e per nulla problematica con le operazioni di saldatura.
Oggi il discorso è totalmente diverso. L'elettronica di consumo è molto più complessa ma molto meno costosa. Si lavora, infatti, esclusivamente con componenti elettronici SMD la cui saldatura richiede particolare abilità. I negozi di elettronica hanno gli scaffali pieni di apparecchiature elettroniche già pronte ma sono sguarniti riguardo la componentistica discreta. E' difficile trovare un singolo integrato o un transistor particolare.
Questa nuova componentistica ha tagliato fuori il 90% degli hobbisti poiché le tecniche e le strumentazioni da utilizzare non sono alla portata di tutti. Del resto questa sorta di sviluppo industriale si rende inevitabile per due fattori di primaria importanza:

1. Le dimensioni dei circuiti elettronici devono essere il più ridotte possibili, in modo da occupare il minore spazio e risultare molto più leggeri e maneggiabili;
2. L'emissione e la ricezione delle alte frequenze (maggiori di 500 Mhz) esige, gioco forza, di superfici estremamente ridotte, per via delle capacità parassite intrinseche nei montaggi in grandi spazi. In pratica non è possibile fisicamente ed elettronicamente realizzare un telefono cellulare o un tablet con componentistica discreta.

Inoltre la complessità circuitale dei moderni dispositivi è tale che è difficilissimo intervenire manualmente a un montaggio o una riparazione. Un solo circuito integrato SMD, dalle dimensioni di mezzo chicco di riso può contenere, infatti, milioni di transistor, ed è letteralmente impossibile stabilire un malfunzionamento se non si hanno costosissime apparecchiature da laboratorio.
Non è possibile, pertanto, costruire in casa telefoni cellulari, tablet, computer, telecomandi ed altro. Tale possibilità è strettamente riservata all'industria. Il povero progettista hobbista deve per forza limitarsi alle realizzazioni molto più semplici.
Oggi, anche da un punto di vista economico, non conviene più prevedere una riparazione di un dispositivo elettronico. Sia all'utente sia al fornitore è più conveniente immaginare la sua completa sostituzione, poiché i circuiti, in fabbrica, costano davvero poco. Se si danneggia un tablet, in garanzia, l'azienda produttrice non ci pensa due volte al rimpiazzo e al ricambio dell'intero articolo piuttosto che alla sua riparazione. Del resto impiegare un operaio che controlli, analizzi e sostituisca i componenti mancanti e danneggiati costerebbe sicuramente molto di più.
Tutti questi fattori hanno decretato una notevole diminuzione degli hobbisti.

I mercati orientali

I paesi orientali dispongono di una tecnologia all'avanguardia. Soprattutto nel ramo elettronico primeggiano da sempre nel mercato. Anche i costi delle realizzazioni elettroniche (di fascia economica) sono estremamente bassi per via di due gravi aspetti:

1. Sono utilizzati componenti elettronici di media-bassa qualità;
2. La manodopera è sottopagata e spesso i lavoratori sono costituiti da giovanissimi che prestano la loro preziosa opera anche per 16 ore al giorno.

Tutti questi fattori hanno fatto sì che il mercato orientale elettronico invadesse il nostro. Oggi le nostre città sono piene di negozi cinesi che mettono a disposizione qualsiasi tipologia di articolo tecnologico. i circuiti elettronici già pronti si vendono a prezzi ridicoli. Con pochissimi euro è possibile trovare preziosi sistemi che, altrove, costerebbero molto di più. Quello che costa meno che si vende di più. E' una dura legge di mercato. Una breve lista di dispositivi, con il relativo prezzo medio, serve per comprendere come per il commercio sia più importante la quantità della qualità:

- Inverter 12V/230V: 20 €;
- Pendrive: 5€;
- Adattatore USB/SD: 3 €;
- Batteria ricaricatore per tablet: 4 €;
- Mouse o tastiere USB: 2 €;

- Tester digitali: 4 €;
- Amplificatori audio: 20 €;
- Radio ricevitore OM, OL, OC, FM, VHF: 6 €;
- E così via.

Se l'hobbista provasse a realizzare una torcia lampeggiante composta da 50 Led ad alta luminosità, che in questi negozi si può trovare anche a 5 €, egli dovrebbe spendere, come minimo, 40-50 €, tra basetta, componenti, Leds, custodie, ecc.
Si tratta, ovviamente, di articoli di media qualità. Perché il progettista o l'hobbista dovrebbe spendere centinaia di euro e "sprecare" tante ore di duro lavoro, quando si possono procurare, per pochissimi euro, tutta questa roba?
E se poi si danneggia qualcosa, non c'è problema: si acquista un nuovo articolo, risparmiando sempre sull'eventuale spesa presso un altro negozio più professionale.

Il mercato

Purtroppo la gran parte di fornitori di materiale elettronico è spesso disinteressata al mondo dell'hobbista. Esistono ancora alcuni negozi, ma molto pochi, con la vera passione di tenere in piedi il mondo della sperimentazione e dell'hobby. I componenti elettronici si troveranno sempre sul mercato. Con Internet oggi è possibile acquistare qualsiasi cosa. Nel negozio sotto casa ovviamente la probabilità di trovare il componente cercato è sempre più bassa. In compenso ci si può rivolgere a grosse aziende di distribuzione elettronica, accollandosi anche le spese di spedizione. Molti negozi e piccoli distributori chiudono battenti, anche per via della crisi, gettando nella spazzatura migliaia di componenti elettronici e centinaia di valvole. Per fortuna queste ultime sono ritornate molto di moda e molti appassionati stanno ricominciando a costruire radio ricevitori e amplificatori audio, per il piacere di ritornare al passato.

Perché costruire da sé i circuiti?

Veniamo, dunque, alla domanda principale, che ha dato vita a questo articolo. Qual è la motivazione per cui un hobbista di elettronica deve continuare (o iniziare…) a costruire i propri circuiti elettronici in casa, con le sue mani? Perché perdere tanto prezioso tempo, col rischio di bruciare tutto, quanto il mercato offre tutto pronto a prezzi stracciati?
Le motivazioni sono tante, esaminiamole approfonditamente e singolarmente.

Passione

L'elettronica hobbistica è, appunto un hobby e come tale va trattato. E', dunque, un'attività praticata nel tempo libero per puro interesse e divertimento. La passione che ne deriva è tanta e chi la pratica è disposto a spendere molti quattrini, pur di veder appagate le proprie soddisfazioni. La ricerca di uno schema non lo spaventa, sfoglia riviste sulle riviste e Internet lo agevola e lo aiuta molto.
E' naturale che se la tecnologia si evolve anche gli hobbisti devono adeguarsi ad essa seguendo il suo naturale corso.

Progetti non disponibili sul mercato

Con l'autocostruzione si può costruire tutto ciò che si vuole e che nel mercato non esiste. Progetti personalizzati e a misura "di utente" possono essere tranquillamente costruiti dall'hobbista. Ricevitori radio, lampeggiatori particolari, strumentazione audio e altro, che il mercato professionale ritiene superfluo, può essere l'oggetto di valide realizzazioni serie e ben funzionanti. E' proprio questo il succo del problema: realizzare ciò che non esiste.
Si tratta, quindi, di una grande occasione da non perdere per veder nascere dal nulla progetti mai visti, originali e unici. Con un po' di fortuna si potrebbe vedere il proprio lavoro candidarsi per una possibile domanda di brevetto.

I software di oggi agevolano il lavoro

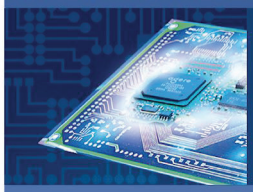
Oggi esistono software di sbroglio PCB e di simulazione elettronica che agevolano enormemente il lavoro dell'hobbista. Tanti anni fa questi strumenti non esistevano e l'intero lavoro di progettazione e realizzazione doveva essere fatto interamente a mano. Non erano rari i casi in cui i propri circuiti esplodessero a causa delle piste costruite male o per un errore di calcolo nel dimensionamento dei componenti elettronici.
Il disegno dello schema era la parte più difficile. Occorreva conoscere bene la legge di Ohm (ma anche oggi è un requisito essenziale) ma fino a quando non se ne realizzava il relativo circuito, non si poteva prevederne il funzionamento. Oggi con i software di simulazione è possibile eseguire qualsiasi calcolo, dimensionamento o, addirittura, un corto circuito. Non ci saranno mai esplosioni di sorta e tutto il lavoro di collaudo avviene esclusivamente sull monitor del computer.
Anche il disegno del PCB era un onere insormontabile la cui difficoltà cresceva logaritmicamente. L'esistenza di errori nel disegno delle piste era un fatto normale e il controllo di qualità era estremamente complesso e laborioso. Oggi, con i software a disposizione, freeware e commerciali, la tracciatura delle piste e lo sbroglio sono un gioco da ragazzi.
L'hobbista prenda, dunque, spunto da tali fattori per iniziare o continuare il proprio hobby. Grazie a ciò egli può dedicarsi ad altre fasi lavorative, come la saldatura, e risparmiare tempo sullo stadio preparativo.
A tale scopo, proponiamo un breve elenco di programmi utili al progettista per le realizzazioni elettroniche. Alcuni software sono freeware, altri a pagamento. La lista non ha alcuna pretesa di completezza ma si limita a elencare i più famosi e importanti.

- **Disegno schemi elettrici e/o simulazione**
    - LTspice;
    - Isis;
    - MMana Gal (progettazione antenne);
    - Aade Filter Design;
    - Circuit Maker;
    - Qucs;
    - Tina Pro;
    - PSpice;
    - Ngspice;
    - sPlan;
    - E molti altri…

THE ORIGINAL SINCE 1994
PCB-POOL®
Beta LAYOUT

PCB prototipi e piccole serie

Servizio puntuale o gratuito
Tempi di consegna a partire da 8 ore



Servizio di assemblaggio
Anche a partire da un solo componente

e-mail: info@pcb-pool.com

www.pcb-pool.com

25 YEARS Beta LAYOUT
create : electronics

PCB-POOL® è un marchio registrato di Beta LAYOUT GmbH

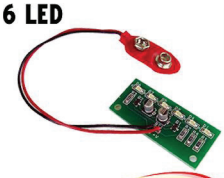
eSTORE®
Beta LAYOUT

Realizzazione, montaggio, saldatura

Iteaduino Leonardo V1.0, compatible

€ 18,60

Lampeggiatore 6 LED



€ 6,00

Frequenzimetro universale, 2,7 GHZ, com interfaccia RS-232C

€ 362,95

Toolkit Extended

€ 149,00

www.beta-eSTORE.com

25 YEARS Beta LAYOUT
create : electronics

eSTORE® è un marchio registrato di Beta LAYOUT GmbH

- **Disegno PCB e sbroglio**
    - Ares;
    - ZenitPCB;
    - FreePCB;
    - Osmond PCB (per Macintosh);
    - ExpressPCB;
    - Kicad;
    - CadSoft EAGLE;
    - Sprint-Layout;
    - E molti altri…

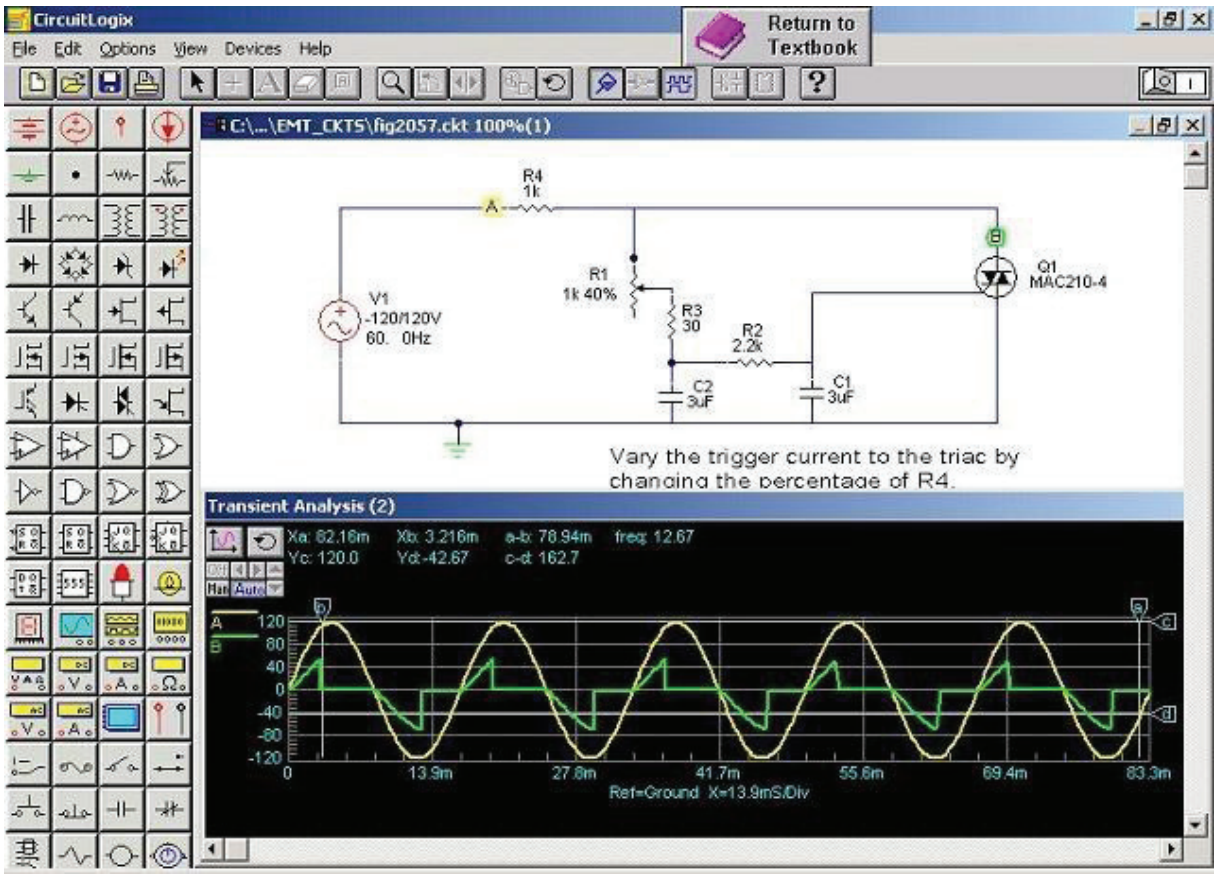


Figura 3: Un software di simulazione elettronica

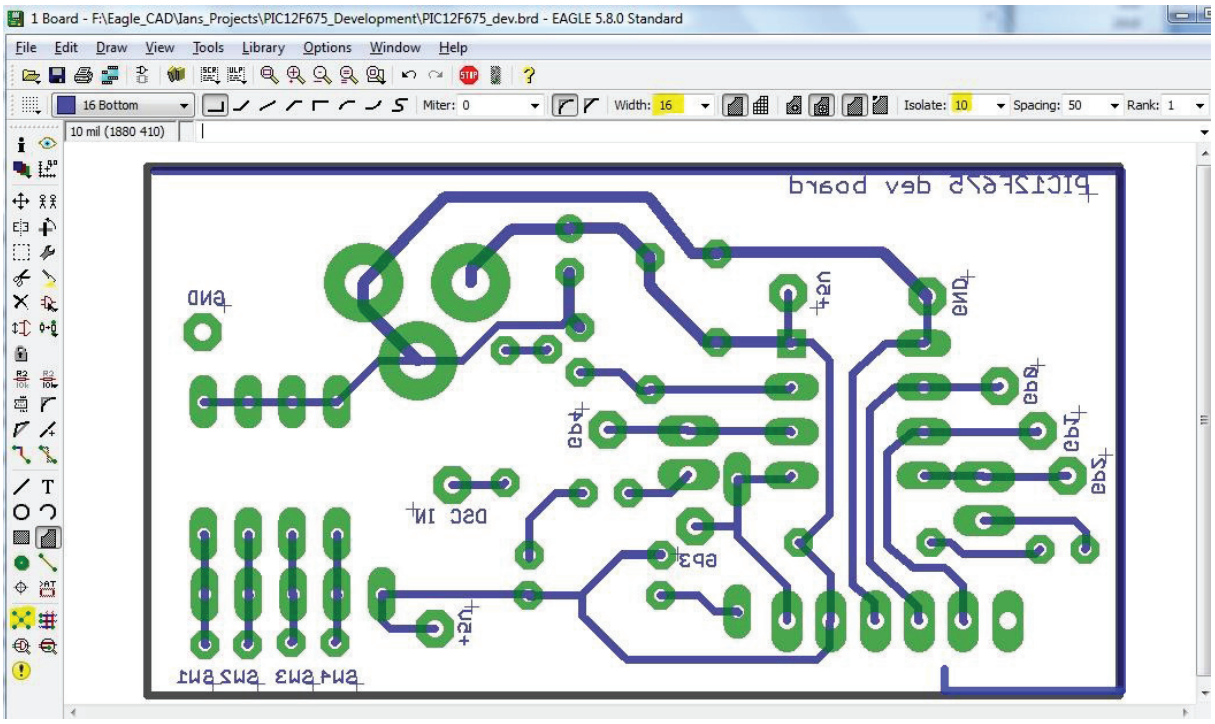


Figura 4: Un software per lo sbroglio dei PCB

Didattica

Con l'autocostruzione mediante componenti discreti si consegue una migliore e approfondita comprensione del circuito. Il motivo è presto detto: ogni singolo componente esplica una determinata e ben precisa funzione elettrica ed elettronica. Seguendo lo schema, l'hobbista può tenere traccia dei segnali e capire l'intero funzionamento del prototipo. E' la stessa cosa paragonare il cucinare una ricetta già pronta in busta e il preparare, passo per passo, la stessa pietanza utilizzando singoli ingredienti.

Un circuito integrato, o addirittura un intero sistema integrato, si presenta invece come una magica scatola nera. Da essa escono solo alcuni pin di connessione e nulla più. Non si comprende quale funzione espleti all'interno, e anche leggendo i relativi datasheet si scopre che esso è composti da migliaia di componenti miniaturizzati. Ovviamente lo spazio occupato è veramente piccolo ma, oltre ad esulare l'hobbista dalla reale comprensione dei vari funzionamenti, lo distoglie dallo studiare e capire la realtà delle cose.

Il progettare da sé i circuiti migliora l'assimilazione dell'elettronica, insegna ad eseguire calcoli matematici, a capire il perché un difetto possa pregiudicar l'intero funzionamento.

In aggiunta, consente di isolare e identificare il singolo componente difettoso, la presenza di piste interrotte, la possibilità di esistenza di saldature fredde e così via.

Essa dona, in pratica, quella sensibilità e quella competenza che nessun libro può regalare. La possibilità di poter sostituire una tipologia di componenti con altri equivalenti è un altro valore aggiunto di questa pratica. Il capire se un transistor può essere rimpiazzato con un altro, osservandone i risultati e valutandone le differenze in termini di funzionamento e di risultato finale.

Sono questi, tutti aspetti positivi, che formano le ossa all'hobbista e sperimentatore. Dopo questa fase, nella quale la formazione teorica e pratica hanno modellato la tecnica e la competenza, si possono utilizzare i circuiti integrati con molta più disinvoltura e sicurezza.

Figura 5: I ragazzi imparavano l'elettronica divertendosi

Potenza di Internet

Oggi tutti noi siamo fortunati sperimentatori, grazie alla presenza di Internet, la grande rete. In passato le conoscenze erano molto ridotte. Scoprire le caratteristiche di un componente elettronico era come seguire una caccia al tesoro. A volte si potevano acquistare costosi datasheets su CD ROM o su supporti cartacei. Ma essi non erano completi, il componente raro e critico non era mai presente e il più delle volte restava nel completo anonimato. Con i motori di ricerca, invece, l'elettronica non ha più segreti. Le informazioni su un dato transistor, su una vecchia valvola e su un intero schema elettrico sono immediatamente scovate. Dati tecnici, caratteristiche elettriche e fisiche, immagini, diagrammi e curve di risposta, non costituiscono più un segreto per nessuno. Si tratta, dunque, di un'arma molto potente che l'hobbista può utilizzare a proprio vantaggio, per risolvere qualsiasi problema.
Forse è proprio questo che sta facendo ridurre la passione dell'autocostruzione. In passato le difficoltà delle realizzazioni erano maggiori ma, al contrario, esisteva una sorta di "magia" che avvolgeva e accompagnava le varie esperienze. Oggi è tutto più facile, scontato e alla portata di tutti.
Con la grande rete anche il reperimento dei componenti elettronici è un gioco da ragazzi. Basta accedere a uno dei tantissimi siti di vendita online, ordinare il prodotto desiderato e aspettare ventiquattro ore l'arrivo del pacco. Materiale che prima si riteneva irreperibile, oggi si trova a bassi prezzi e facilmente, anche richiedendoli ai mercati molto distanti e diversificati dal nostro.

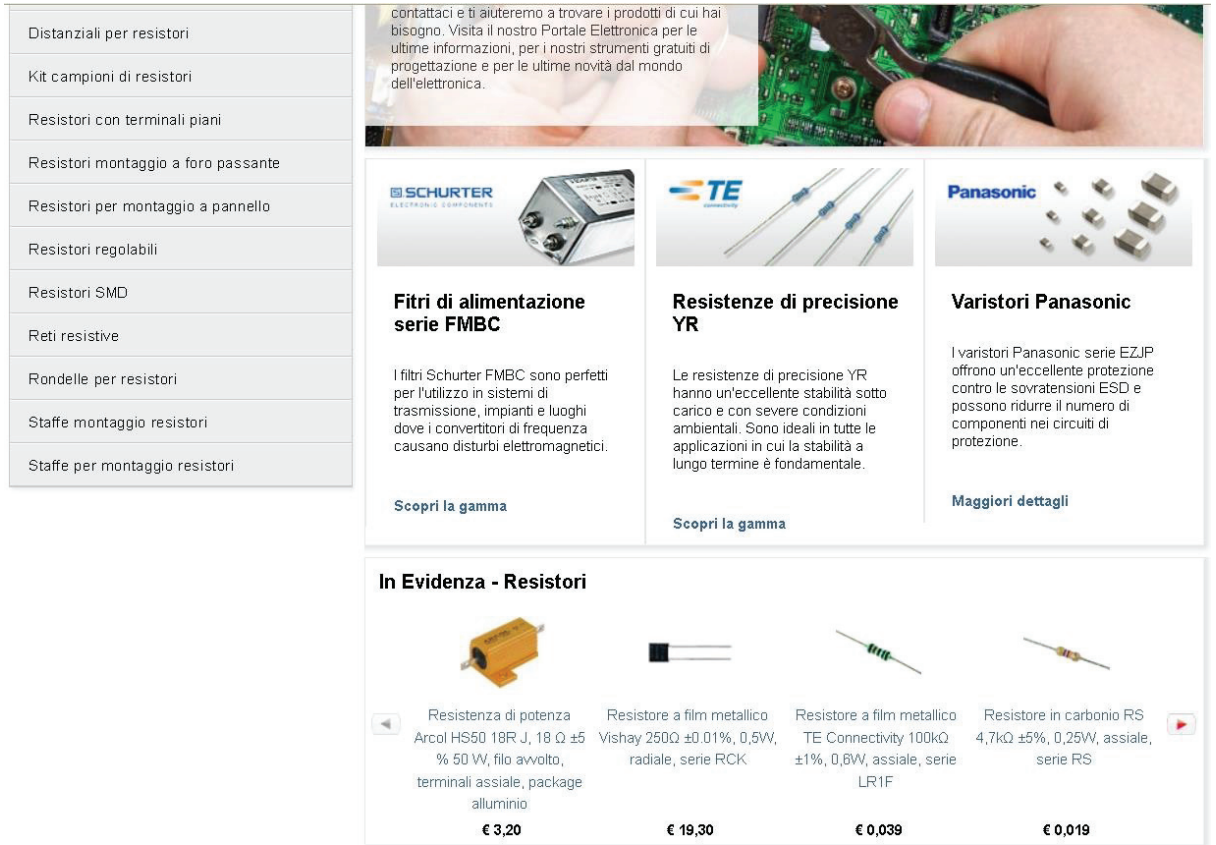


Figura 6: Un sito per l'e-commerce di componenti elettronici

Presenza dei microcontrollori

In passato, la costruzione di un complesso lampeggiatore programmabile dall'utente richiedeva circuiti estremamente complessi. Bisognava utilizzare soluzioni complicate, colme di porte logiche, flip flop, temporizzatori e circuiteria discreta a iosa. Sì, è vero, la progettazione di questi sistemi era molto complicata. Oggi basta un semplice microcontrollore (MCU) a 8 o 16 bit e un piccolo firmware, scritto ad hoc con un linguaggio di programmazione, per realizzare facilmente il prototipo. Sono, in pratica, diminuite le fasi lavorative sul circuito elettrico ma sono aumentate le procedure di programmazione e automazione del software. Il progettista, oggi, è anche un analista e un programmatore informatico. E tutto questo fa paura a molti amatori della materia. Il 90% degli hobbisti, con l'avvento dei sistemi automatici e programmabili, si sono, di colpo, fermati e bloccati. Hanno reputato invalicabile l'ostacolo della programmazione.

Ovviamente un microcontrollore, perfetto in tutte le sue funzionalità, toglie tutta la magia di un circuito. E' una super scatola nera nella quale avviene tutto. Addirittura, riprogrammandolo, e senza toccare minimamente il circuito elettrico, può rispondere in maniera totalmente diversa agli eventi. Il passo dei tempi è stato segnato e l'appassionato deve, gioco forza adeguarsi, pur continuando a usare le vecchie tecnologie.

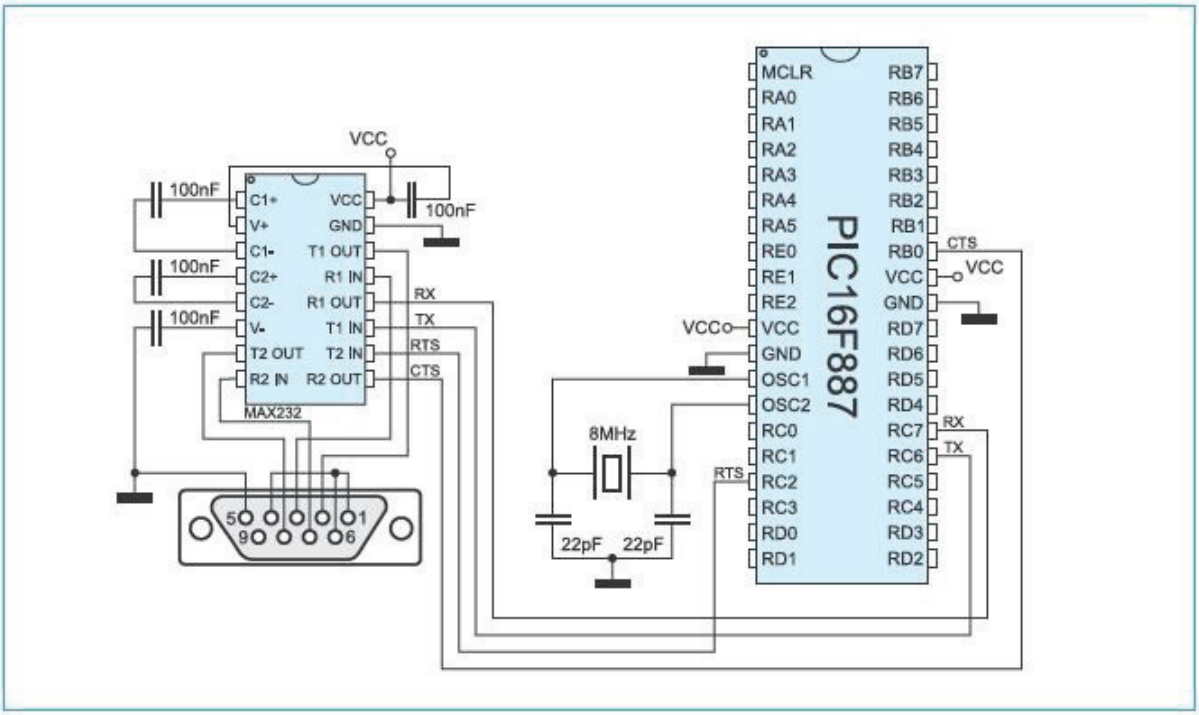


Figura 7: Con un microcontrollore si realizzano circuiti estremamente sofisticati

La figura dell'hobbista oggi

L'appassionato e l'hobbista elettronico di oggi sono sostanzialmente diversi da quelli di ieri. Oggi essi "smanettano" molto meno con i componenti elettronici. Forse usano il saldatore un po' meno e non riducono più "in fumo" alcun transistor. Simulano prima il circuito con tanta attenzione con i software adeguati e demandano la preparazione dei PCB ad aziende specializzate che, con una cifra davvero esigua, preparano circuiti stampati molto professionali e pronti all'uso (persino i doppia faccia).

L'hobbista di oggi usa sistemi embedded, schede elettroniche già pronte all'utilizzo e confezionate, microcontrollori programmati e prototipi funzionanti. Impiega motori passo passo, robot, moduli ibridi e microcomputer. Si adopera su un assemblaggio molto generalizzato su parti pronte e subito adoperabili.

Coloro che sanno impugnare il saldatore sono molto pochi e probabilmente non ce n'è nemmeno bisogno, giacché tutto il sistema è già pronto.

Ci s'interessa maggiormente dei componenti SMD ma coloro che sanno utilizzarli sono in numero limitato. Specialmente chi ha una certa età, comincia a odiare queste soluzioni elettroniche, poiché anche i problemi della vista cominciano a prender piede.

Differenze tra gli hobbisti di ieri e quelli di oggi

La tabella che segue illustra, a grandi linee, le differenze sostanziali tra gli appassionati di elettronica di ieri e di oggi. Per "ieri" s'intende un periodo remoto di circa quaranta anni fa, quando non esistevano le comodità informatiche e tecnologiche di oggi.

	Hobbisti del 1980	Hobbisti del 2014
Saldatore	Classico saldatore da 30W.	Stazione saldante e dissaldante.
Microcontrollore	Non si utilizzavano. I sistemi decisionali erano realizzati con decine di porte logiche, producendo circuiti molto ingombranti.	Con una piccola MCU si possono approntare circuiti ridotti ma molto complessi.
Onde radio	Gli appassionati realizzavano in casa radio ricevitori di tutti i generi, specialmente nella gamma delle onde corte. Antenne e bobine auto costruite erano all'ordine del giorno.	Gli appassionati hanno quasi del tutto abbandonato tale pratica. Le comunicazioni tra punti vicini avvengono tramite Wi-fi e a grande distanza tramite Internet.
Circuiti Integrati	Si utilizzavano spesso gli integrati DIL, a passo 2,54 mm molto comodi da usare e da sostituire.	Gli SMD sono ormai entrati nella prassi normale della progettazione. Non si usano solo circuiti integrati ma anche resistenze, transistor e altra componentistica discreta a passo estremamente ridotto.
PCB	Normalmente si realizzavano circuiti stampati semplici, con piste larghe e monofaccia. Si usava la tecnica del pennarello indelebile o la fotoincisione (da parte dei più bravi). La progettazione del disegno delle piste avveniva manualmente con alta probabilità di errore finale.	I software di sbroglio oggi rendono la vita estremamente semplice. Dallo schema, in maniera del tutto automatica, essi riescono a generare i disegni per un circuito stampato, anche estremamente complesso. I circuiti doppia faccia e multi-layer sono utilizzati e possono essere realizzati con diverse tecniche.
Simulazione	Prima non	Con un programma di

	esistevano programmi di simulazione analogica e digitale. Doveva essere calcolato tutto manualmente, tramite formule matematiche, anche complesse, col rischio di non indovinare sempre tutto. Se per l'elettronica digitale poteva essere relativamente semplice, per quella analogica sorgevano insormontabili problemi di previsione. I circuiti dovevano essere sempre realizzati per provarne il loro funzionamento e se un errore di calcolo impediva il buon funzionamento, tutta la realizzazione poteva finire in fumo.	simulazione è un gioco da ragazzi disegnare lo schema elettrico, sia esso analogico, digitale o promiscuo. Si possono visualizzare facilmente i segnali e le tensioni presenti su ogni nodo, disegnare grafici e sostituire, in tempo reale, componenti e modificare in maniera semplice valori e parametri. Si elimina, in tal modo, la possibilità di ridurre tutto in fumo e di prendere qualche pericolosa scossa elettrica.
Assemblaggio provvisorio	I componenti si assemblavano su configurazione "volante", specialmente per i ricevitori radio. Si utilizzavano anche primordiali basette millefori.	Oggi si utilizzano le Breadboard o le basette millefori.
Da chi acquista i componenti?	Esclusivamente presso il negozio sotto casa.	Tramite Internet e i rivenditori specializzati, si trova praticamente qualsiasi componente elettronico, anche non più prodotto o particolarmente raro.
Documentazione	La documentazione e i datasheet erano un po' difficli da trovare. In	Oggi esiste Internet. Una rapida ricerca su Google e si trova qualsiasi informazione.

	commercio esistevano costosi CD pieni di documentazione e datasheet in formato PDF. Ma spesso erano incompleti e mancavano le informazioni sui componenti elettronici più rari. Il negoziante, al massimo, aveva grossi cataloghi cartacei che riportavano le caratteristiche tecniche e modelli di equivalenza.	

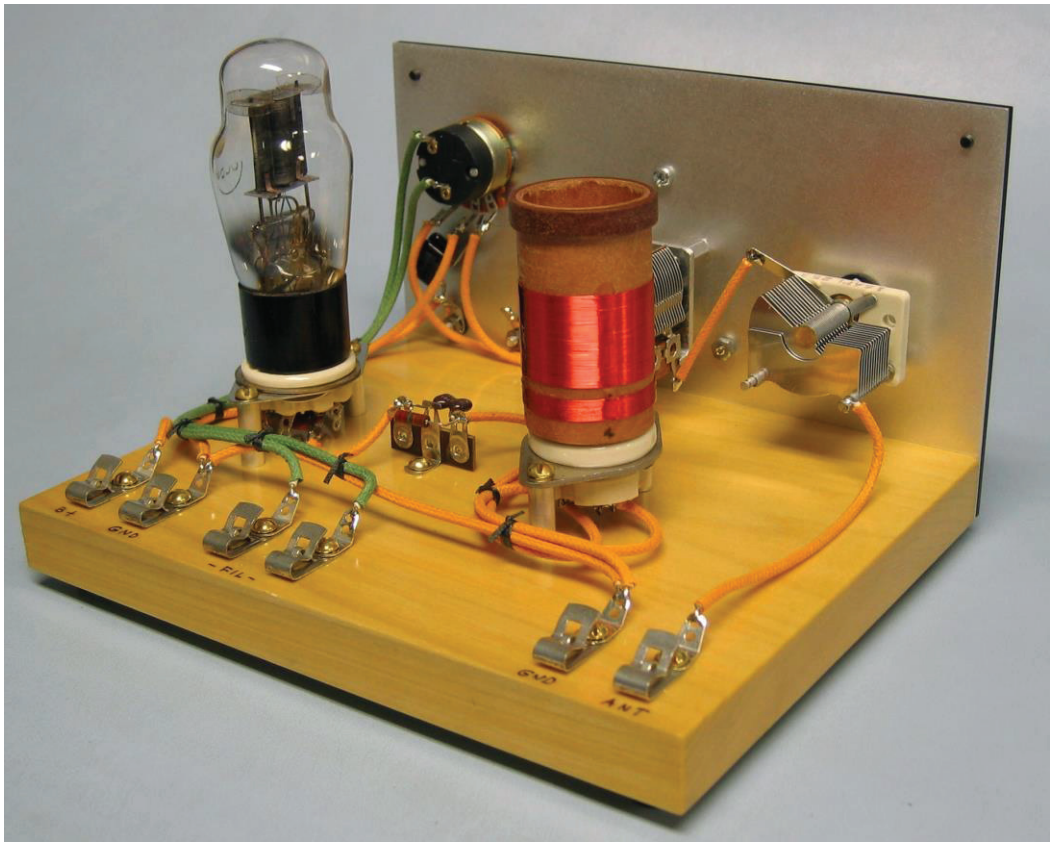


Figura 8: Ricevitore radio a valvola autocostruito

Figura 9: Prima l'hobby dell'elettronica coinvolgeva anche i più giovani

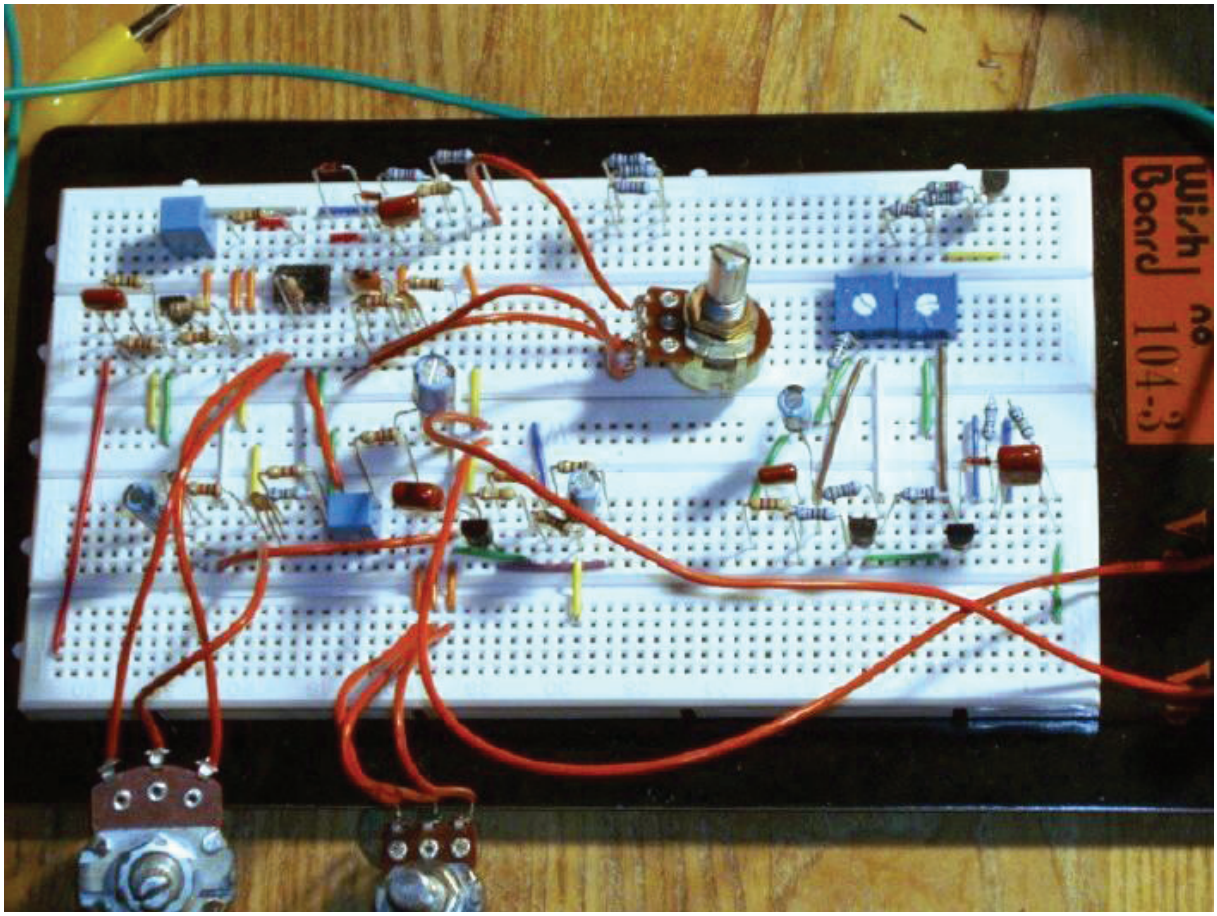


Figura 10: Oggi la BreadBoard aiuta l'hobbysta

Conclusioni

La passione hobbistica per l'elettronica è un'attività che, certamente, non scomparirà mai. Esisteranno sempre gli appassionati che, seppur in numero ridotto, conserveranno per sempre l'amore e la dedizione per questo fantastico esercizio ricreativo e didattico allo stesso tempo.
E la magia che si riscopre nel saldare i componenti sul proprio PCB e collaudare i propri circuiti funzionanti non ha eguali.
Magari alla fine si resterà con il proprio laboratorio a soqquadro e con qualche componente elettronico in più bruciato ma pienamente appagati e soddisfatti del proprio lavoro.

FORMAZIONE CONTINUA PER SVILUPPARE IL CAPITALE UMANO E FAVORIRE LA COMPETITIVITÀ

NON BUTTARE VIA I TUOI SOLDI!

RECUPERA UNA PARTE DI VERSAMENTI INPS E **INVESTILI IN FORMAZIONE** DEL PERSONALE, GRAZIE ALL'**INNOVATIVO PROGETTO TECNOIMPRESE**

- Ogni impresa può destinare lo 0,30% dei contributi INPS a un Fondo interprofessionale con un meccanismo simile al 5 x 1.000 della dichiarazione dei redditi, (non si tratta quindi di un costo aggiuntivo);
- Tecnoimpreser assiste le aziende nella ricerca di agevolazioni per la formazione dei dipendenti, valutando la possibilità di accedere ai finanziamenti pubblici;
- Tecnoimprese, grazie a un accordo siglato con FormAzienda (fondo autorizzato dal Ministero del Lavoro) ha in programma una serie di seminari **GRATUITI** in tutti il Nord Italia

Tecnoimprese è iscritto all'albo enti di formazione della Regione Lombardia

FORMAZIONE TECNOIMPRESE

Tecnoimprese: Via Console Flaminio, 19 - 20134 Milano
tel. 02 210.111.230 - *training@tecnoimprese.it*

www.tecnoimprese.it/formazione

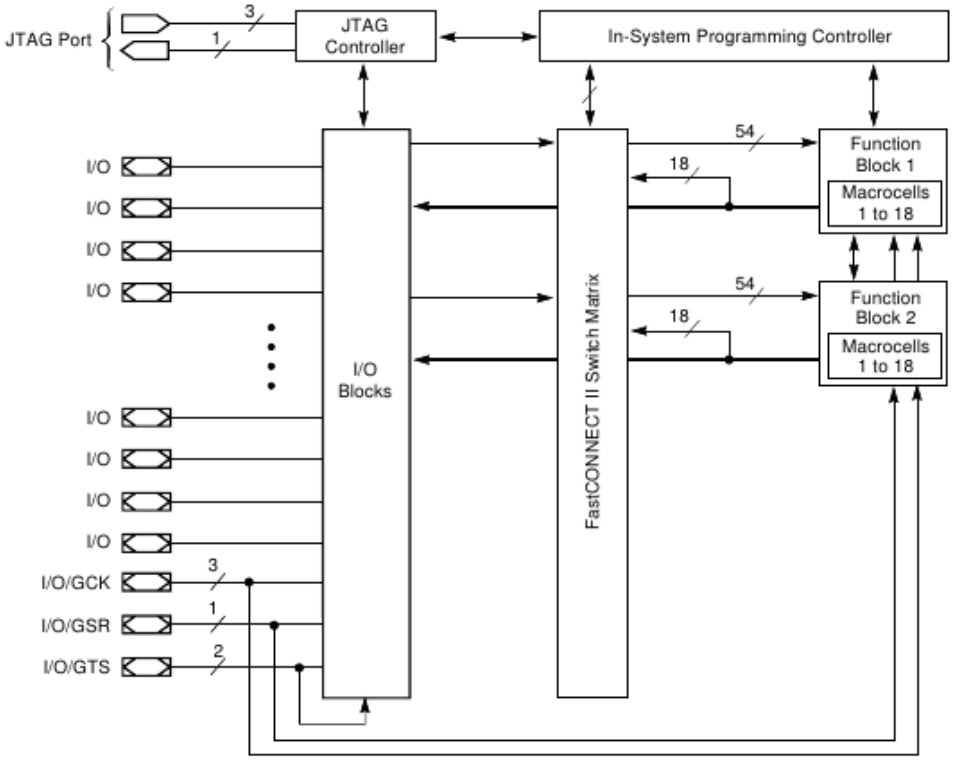
LA FPGA BOARD

Di Massimiliano Miocchi

Nella precedente pubblicazione abbiamo affrontato sommariamente le differenze tra CPLD ed FPGA, nonché il concetto del termine Asics. Bene ora è venuto il momento di parlare della FPGA board, la scheda che ci consentirà di fare pratica cercando di capire ed imparare a maneggiare questa interessante tecnologia. In questa sezione affronteremo in particolar modo alcuni blocchi funzionali a corredo della FPGA cercando di capirne il perché e il loro funzionamento.
Ci cimenteremo in applicazioni come ormai molti hardware fanno, dall'accensione di un diodo led alla visualizzazione di numeri o altro su display alla gestione degli input, in questo mio progetto ho apportato delle leggere modifiche alla consuetudine e precisamente ho aggiunto oltre che i soliti display e led anche un ADC con trasmissione SPI, questo per cominciare a far pratica con protocolli seriali non sempre compresi, un potentissimo processore in configurazione ADC code per semplificarvi un pochino la gestione di sensori di temperatura, pressione e altre specie.

Fpga XC9536xl

Il cuore del sistema è comprensibilmente la CPLD XC9536XL, la scelta ricade su un modello di Cpld poco costosa e di facile reperibilità e con un discreto numero di moduli primitivi sufficienti al nostro scopo. Ci tengo a dire a tutti i lettori che nulla cambia concettualmente nell'utilizzare Cpld o Fpga , basta tenere sempre a mente le regole da seguire in fase di progettazione. In questo paragrafo seguiremo un concetto schematico che ci aiuterà nella costruzione delle FPGA Board. Cominciamo col dire che la CPLD in oggetto è cosi costituita:



come si denota dallo schema questa CPLD veloce ha diversi I/O puri le connessioni per la programmazione e la reading in the circuit costituite dai pin:

TMS
TCK
TDI
TDO
VCC
GND.

Un aspetto importante di questi straordinari dispositivi della Xilinx è la gestione contemporanea di tre fonti di clock, si osservino i pin GCK1,GCK2,GCK3 questo da la possibilità di realizzare sistemi dove alcuni processi avvengono in contemporanea e non in sequenza, come accade nei microcontrollori. Il vantaggio delle FPGA ma come le CPLD è la possibilità di programmare architetture diverse che viaggiano contemporaneamente, gestendo in modo simultaneo informazioni comuni, capite che questo è un estremo vantaggio in termini di velocità e performance hw.

SCHEMA A BLOCCHI

Per permettere a tutti gli appassionati di elettronica amatoriale di poter realizzare con semplicità progetti utilizzando CPLD ho cercato di implementare nella board circuiti funzionali che permettono di agevolare alcune funzioni. Per cominciare ho inserito un adc reaizzato con un microprocessore attiny85 con uscita dati SPI che potrà essere letta dalla CPLD , a questo proposito fare attenzione a non impostare i pin DS, SH_CP, e ST_CP come uscite poiché si rischierebbe di danneggiare gravemente la CPLD . Questi pin devono essere settati come ingressi per poter leggere il valore della conversione ADC. Vien da se che per molte applicazioni di un certo livello occorre utilizzare delle FPGA come la **spartan3** anche a performance superiori, ma i prezzi sono proibitivi per uno scopo didattico, quindi la CPLD in questione è più che sufficiente per cominciare a famigliarizzare con questa tecnologia. In questo progetto è possibile utilizzare anche la XC9572 perfettamente equivalente nell'inserimento del socket, la prima ha 32 macrocelle la seconda ben 72. Ricordo anche a tutti i lettori che le CPLD come le FPGA vengono utilizzate per creare HW dedicati dove non gira un fw come nei microcontrollori, il cui linguaggio per eccellenza è il verilog e vhdl. Ben noto il Verilog è un linguaggio di descrizione hardware che permette l'implementazione di sistemi digitali in un ampio spettro di livelli di astrazione: da quello logico a quello architetturale. Esso include, inoltre, costrutti gerarchici che permettono al progettista di controllare facilmente la complessità del sistema.
L'elemento base per la descrizione di un blocco in Verilog è il modulo. Quest'ultimo rappresenta un costrutto logico preso in uno dei suoi possibili livelli di astrazione. Le informazioni base contenute nel modulo sono: un'interfaccia, dove vengono dichiarati gli ingressi e le uscite, e il corpo, dove è sintetizzata la concreta descrizione funzionale del modulo stesso.
L'interfaccia di un modulo è divisa in due parti:

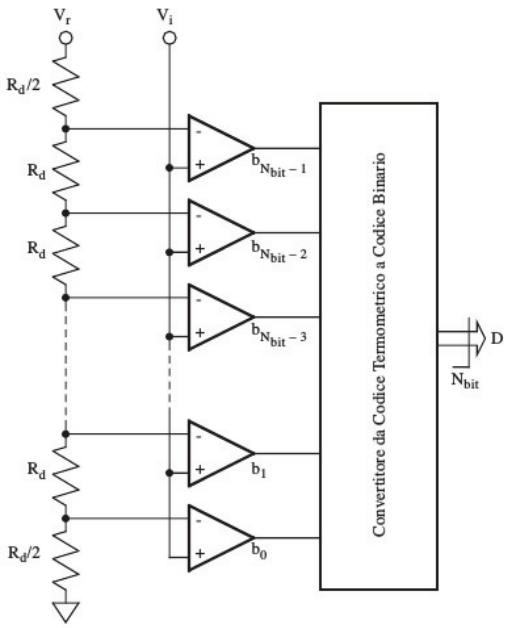
1. Lista delle porte: le porte sono i terminali di ingresso e uscita del modulo. L'elenco delle porte compare fra parantesi subito dopo il nome del modulo. Nella lista, le porte di input/output possono comparire in qualsiasi ordine;

2. Dichiarazione delle porte: subito dopo la prima riga devono essere definite tutte le porte che compaiono nella lista. Per ogni porta si definisce la direzione (input, output, inout), la dimensione (il numero di bit associati alla porta) e il tipo (wire, reg, ecc.). E' importante definire il giusto tipo per ogni variabile in gioco sia essa un ingresso, un'uscita o una variabile ausiliaria interna utilizzata per l'algoritmo. Per com'è concepito, il Verilog viene spesso considerato più semplice e più vicino all'hardware di quanto non lo sia il VHDL; infatti può essere utilizzato anche per modelizzare circuiti logici a livello di transistor o interruttori, cosa che risulta particolarmente difficile con il VHDL. Per contro, il VHDL ha dei costrutti ad alto livello e delle tipologie astratte che il Verilog non possiede, rendendo il VHDL molto più adatto alla descrizione comportamentale. Molti programmi di simulazione e di sintesi RTL accettano sia VHDL che il Verilog, ed è quindi possibile utilizzare il VHDL per la progettazione ad alto livello e il Verilog per la temporizzazione a basso livello post-sintesi. Fatta questa premessa possiamo tornare alla descrizione di uno dei tanti blocchi funzionali inclusi nella board.

CONVERTITORE ADC servito con Attiny85.

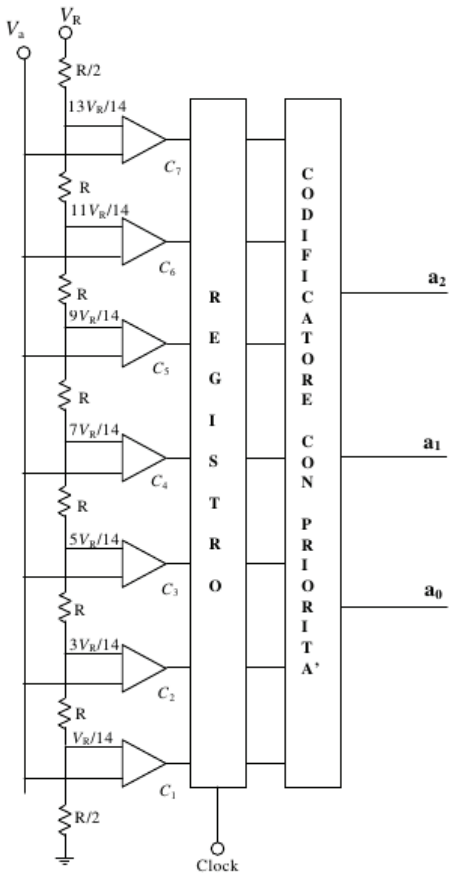
In un sistema ad hardware dinamico non poteva mancare un ADC, se pur assistito da un micro serie Attini85 . Il micro provvede alla conversione da analogico a digitale e trasmette il byte via spi, che a sua volta dovrà essere letto dalla CPLD. Data la complessità per i novizi di realizzare un convertitore A/D totalmente hardware ho preferito utilizzare un adc software da interfacciare alla cpld, nonostante ciò il lettore saprà già che un convertitore AD è possibile realizzarlo ricorrendo ad un comparatore e un contatore binario, proprio come da figura:

Convertitore A/D Flash

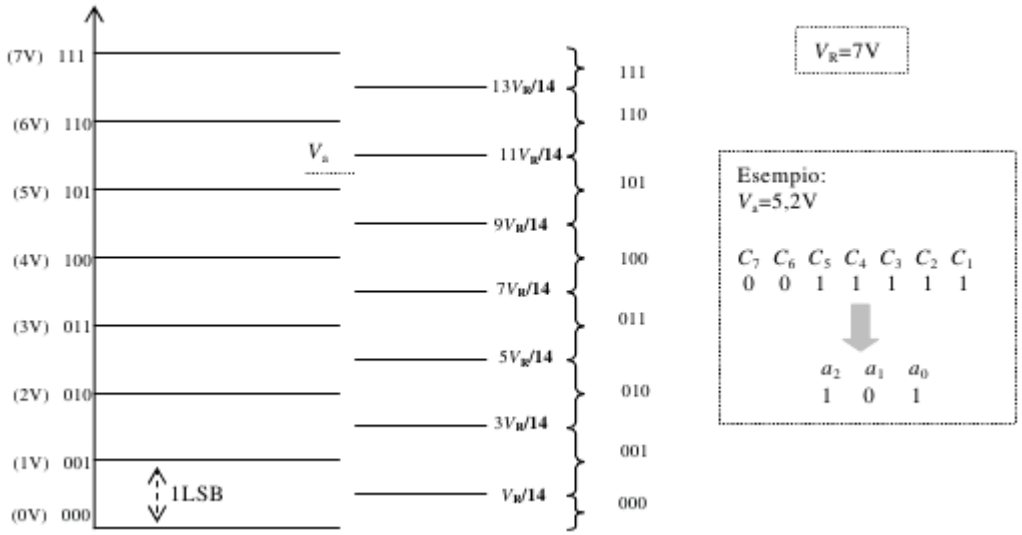


questo tipo di convertitore AD è largamente utilizzato, quindi realizzarlo sarebbe stato complesso e avremmo dovuto comunque aggiungere un comparatore esterno. L'hobbista tenga conto che una CPLD ha risorse minori ad una FPGA sia in termini di macrocelle che di blocchi funzionali, realizzare un convertitore AD porterebbe ad un eccessivo uso delle risorse della CPLD rendendo difficile un approccio semplice e in pieno stile hobbistico.

Il principio di funzionamento è basato sul confronto tra il valore del campione da convertire e i potenziali presenti sui morsetti invertenti dei comparatori dati dalla partizione di una tensione di riferimento VR. In ogni comparatore, sul morsetto non invertente, è presente il valore di tensione da convertire: l'uscita è alta per il comparatore in cui si verifica che Vi è maggiore del potenziale del morsetto invertente. Come risultato si avrà un gruppo di comparatori con uscita al livello alto ed un altro gruppo con uscita al livello basso. Le uscite dei comparatori vanno in un ENCODER che fornisce così il codice binario.
Per fare in modo che tutti i dati arrivino contemporaneamente all'encoder ed evitare disturbi (glitch) si possono inserire dopo i comparatori dei latches. Questi mantengono l'informazione e la presentano simultaneamente all'encoder (si pone un registro a monte dell' Encoder). Si noti che le situazioni che vedono la tensione incognita inferiore alla tensione di confronto minima (uscita di tutti i comparatori al livello basso) oppure superiore alla tensione di confronto massima (uscita di tutti i comparatori al livello alto) devono essere considerate come "non valide" in quanto potenzialmente fuori campo di misura !
A tal proposito possiamo modificare lo schema come da figura:



Fonti di errori sono le soglie di riferimento e i comparatori. Le variazioni dei valori delle soglie sono dovute alle variazioni di VR, che danno luogo ad un errore di guadagno, e a valori disuguali delle resistenze, che determinano una non-linearità della caratteristica. Gli errori introdotti dai comparatori sono dovuti alla corrente di polarizzazione degli ingressi, che carica il partitore e determina errori di non-linearità, e alla tensione di offset d'ingresso che determina uno spostamento della soglia diverso per ciascun livello. I convertitori flash sono usati quando è necessaria la massima velocità di conversione, che può raggiungere valori dell'ordine di 108-109 conversioni/secondo. Per realizzare un convertitore ad N bit, sono necessari 2N-1 comparatori e 2N resistori. Pertanto l'area richiesta cresce rapidamente all'aumentare della risoluzione. I convertitori flash hanno tipicamente una risoluzione minore o uguale a 10 bit. Un esempio di come avviene la conversione è schematizzato nella figura seguente:



e la tabella della verità logica è cosi strutturata:

CODIFICATORE CON PRIORITA'

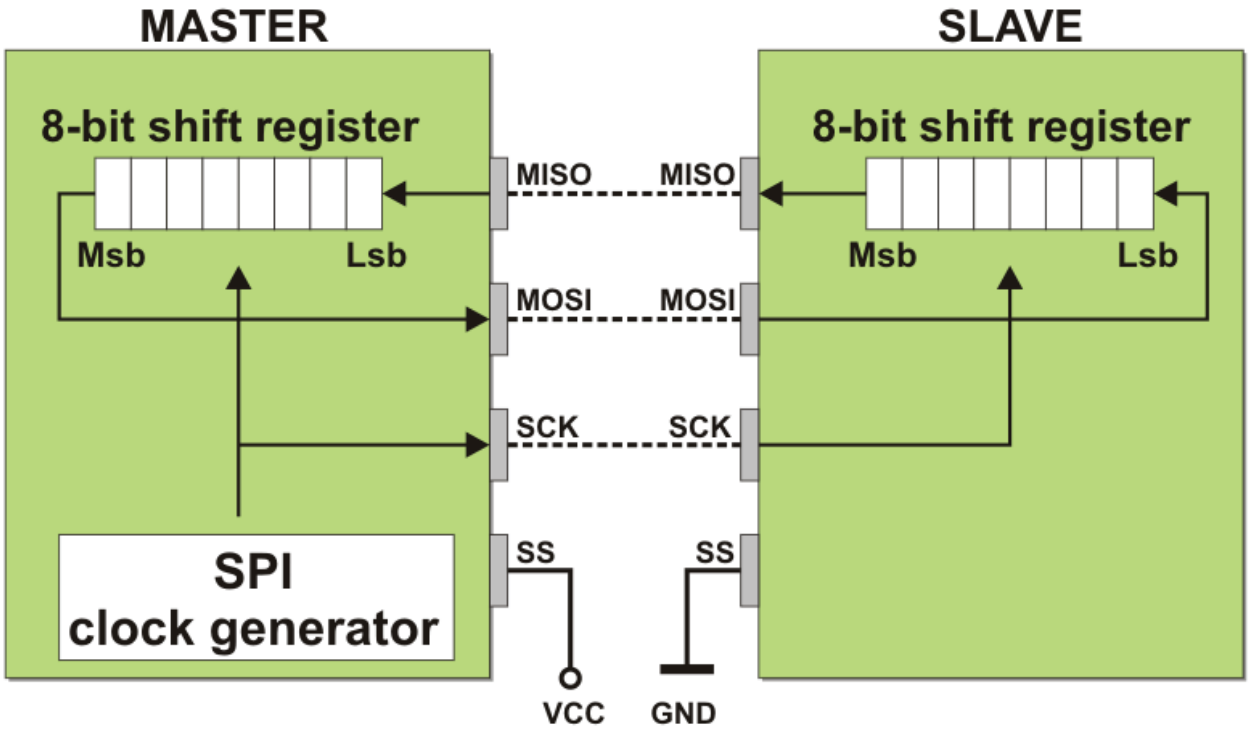
C_7	C_6	C_5	C_4	C_3	C_2	C_1	a_2	a_1	a_0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	1	x	0	1	0
0	0	0	0	1	x	x	0	1	1
0	0	0	1	x	x	x	1	0	0
0	0	1	x	x	x	x	1	0	1
0	1	x	x	x	x	x	1	1	0
1	x	x	x	x	x	x	1	1	1

$x \in \{0,1\}$

Si osservi l'esempio di conversione di una tensione pari a 5.2V, osservando il comparatore con le soglie abbiamo che il valore dell'uscita dei comparatori C7 e C6 sono a zero dato che il valore è inferiore alla loro soglia, per tutti gli altri a uno poiché superiore. Osservando la tabella della verità ci ritroviamo una sequenza di bit che

decodificati riportano in uscita il valore **1 0 1** corrispondente a 5V. Questo per far capire che una FPGA può divenire una rete complessa sequenziale/combinatoria tanto da sostituire microcontrollori e altri dispositivi di calcolo. Le CPLD al contrario essendo molto meno efficaci coprono quella parte di reti combinatorie e sequenziali ove una FPGA sarebbe troppo onerosa oltre che sprecata. Il bello di utilizzare queste meraviglie della tecnologia è la consapevolezza di poter realizzare sistemi di hardware dinamici modificabili senza dover ridisegnare lo schema, sono come camaleonti si adattano molto bene alle situazioni più svariate. Ora data la complessità per molti di realizzarlo a livello hardware ho interfacciato un micro della serie Attiny85 programmato per convertire un segnale analogico in digitale e trasmetterlo in formato seriale SPI. Quello che occorre fare è semplicemente leggere il dato trasmesso e gestirlo di conseguenza riportandolo sui led o sul display.
I segnali preposti al dato in uscita dal micro sono:

DS – dato trasmesso in formato seriale di 8 bit SH-CP e ST-CP tipici del protocollo SPI, questi segnali sono uscite del micro e quindi ingressi della CPLD, in atto di programmazione occorre tenerne conto per evitare di cortocircuitare un output con un input generando guasti alla CPLD. Una tipica trasmissione SPI è rappresentabile come da figura seguente:



dove è ampiamente visibile che il primo bit a uscire sarà l'ultimo a essere letto.

PROTOCOLLO SPI

La acronimo SPI, che sta per ***Serial Peripheral Interface***, si riferisce ad una periferica di interfaccia con caratteristiche seriali. Nella comunicazione SPI sono presenti un unico dispositivo principale detto "**MASTER**" e un certo numero di dispositivi secondario detti "**SLAVE**" connessi al **MASTER** mediante un bus a 4 fili su

ci viaggiano i segnali necessari alla comunicazione SPI trai i vari dispositivi. L' **SPI** usa i seguenti segnali per serializzare lo scambio di dato con un altro dispositivo:

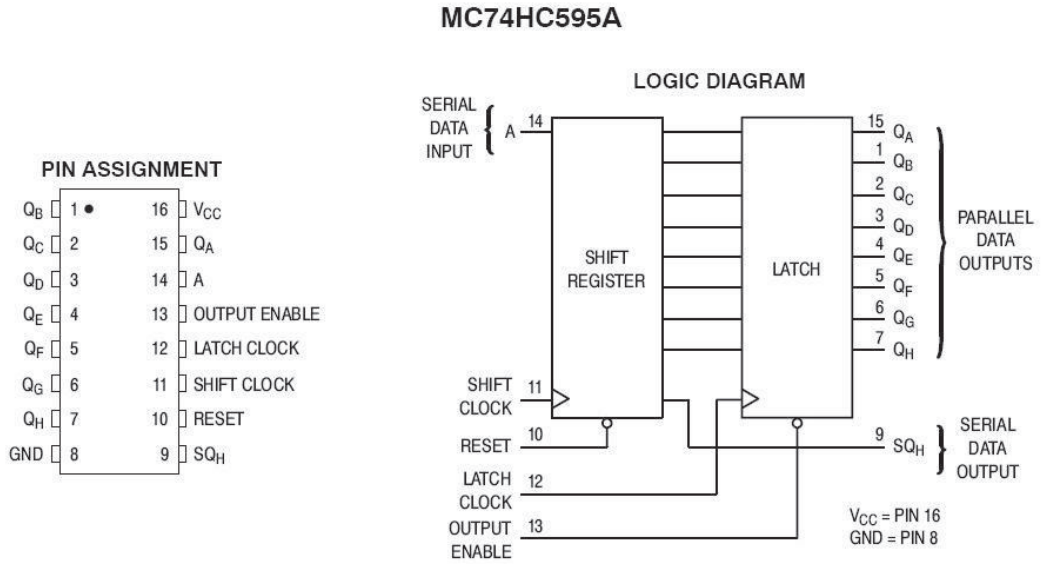
SS - Questo segnale è conosciuto come "SLAVE Select ". Quando passa a livello basso, il dispositivo slave si mette in attesa pronto ad ascoltare il segnali di clock e i dati.

SCK - clock seriale generato dal MASTER che controlla l'invio e la ricezione dei dati

SDO (MOSI) - linea di uscita seriale dei dati di uscita da un dispositivo all'altro.

SDI (MISO) - linea di ingresso seriale dei dati in ingresso ad un dispositivo SPI

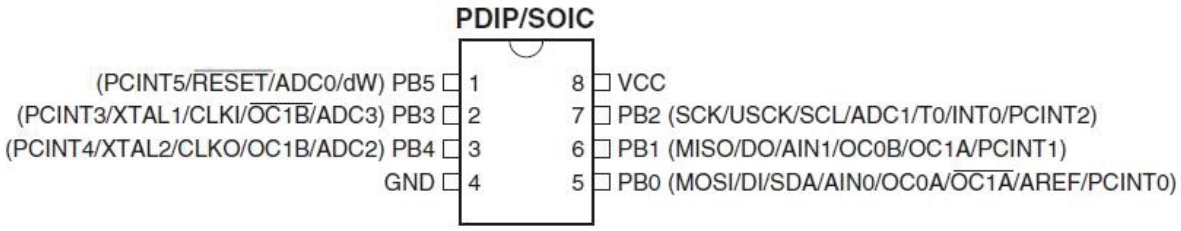
I dati sono "scambiati" sempre fra i vari dispositivi collegati al bus. Nessun dispositivo può essere solo un trasmettitore o solo un ricevitore nel protocollo SPI. Ogni dispositivo ha due linee di dati, una di ingresso **SDI** e una di uscita **SDO**. Lo scambio di dati e' scandito da un segnale di temporizzazione detto clock presente sulla linea denominata con**SCK.** Il clock e' generato dal **MASTER** che, tramite esso, sincronizzazione gli eventi, controlla quando i dati possono cambiare in uscita e quando possono essere letti. Soltanto il dispositivo **MASTER** può controllare la linea di clock **SCK.** Questo e' un vantaggio perché il clock può variare senza distruggere o corrompere i dati. La frequenza dei dati (data rate)cambierà semplicemente con i cambiamenti del clock. Ciò rende l' **SPI** ideale anche quando si ha un clock impreciso come nel caso in cui si utilizzi un oscillatore RC. Realizzare uno SLAVE è esattamente ciò che bisognerà fare con la CPLD, realizzare quindi un hardware che ci consenta di processare i dati in uscita dal micro, in poche parole occorre realizzare una sorta di 74595 o meglio uno shift register a 8 bit esattamente come in figura:



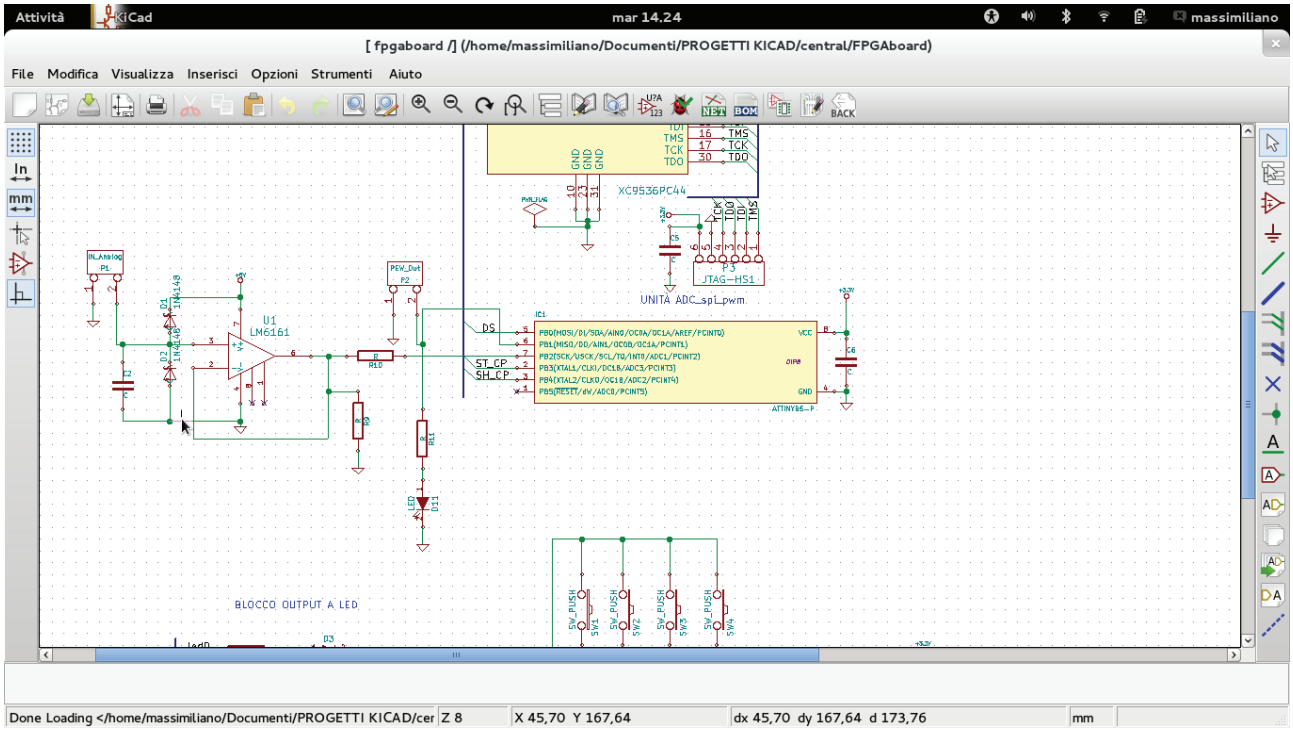
il nostro adc è stato realizzato con attiny85, un micro con un dip 8 dalle grandissime potenzialità, è stato programmato come convertitore adc con trasmissione in uscita dei dati secondo il formato SPI.

1. Pin Configurations

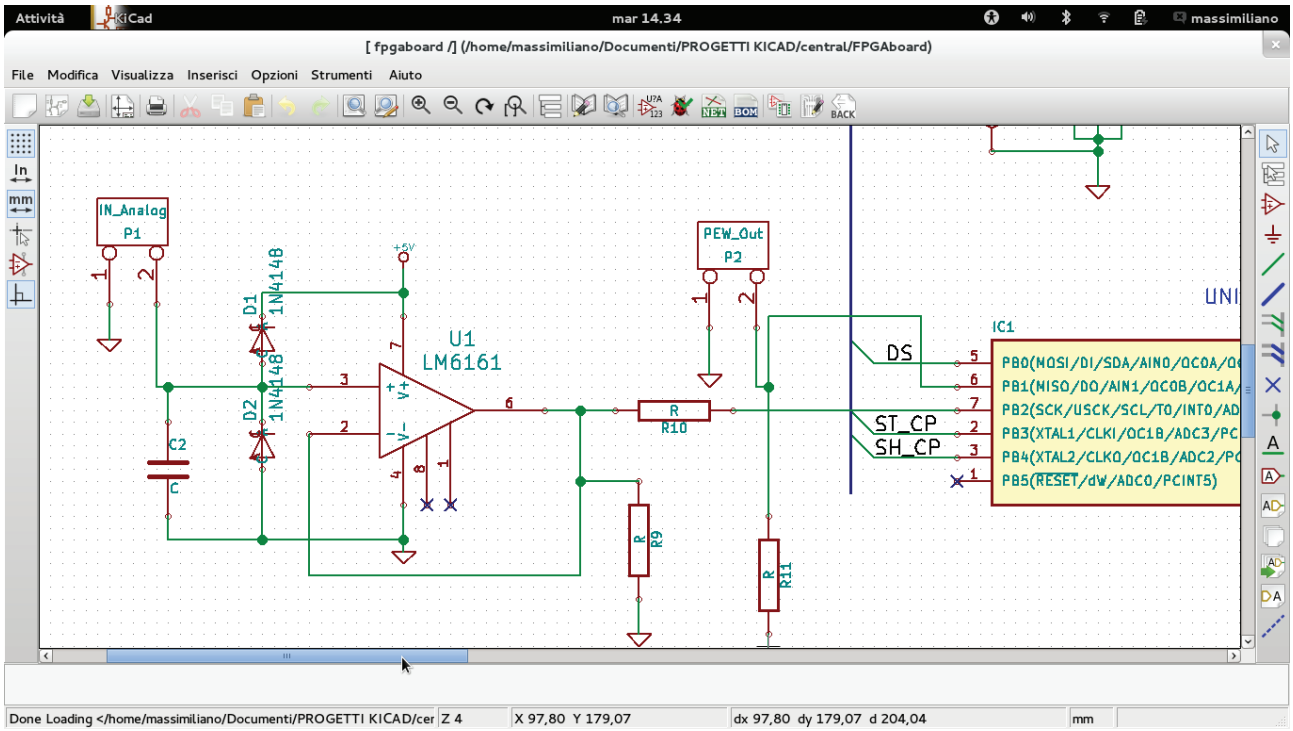
Figure 1-1. Pinout ATtiny25/45/85



La figura mostra la sezione relativa all'adc dove è evidente il segnale DS ,Serial Data, e gli atri due segnali uno per il clock e l'altro per il latch. Ho praticamente replicato il collegamento come se al posto della CPLD o FPGA qual si voglia ci fosse un 74595 shift register. L'operazionale LM6161 sull'ingresso dell'ADC svolge la funzione di adattamento di impedenza e protezione dell'ingresso adc del micro. Il connettore P2 è un uscita PWM che è predisposta su richiesta che nella versione base non è attiva.



ZOOM della sezione adc.



OSSERVAZIONI

Come accennato all'inizio di questo articolo, ho preferito utilizzare una CPLD al posto di una FPGA SPARTAN 3 dato il costo oneroso per molti degli hobbisti, certo la CPLD consente di operare limitatamente poiché la XC9536 o la XC9572 possiedono pochi gate e poche macrocelle, ma questo limite è stato parzialmente superato utilizzando l'apporto dei micro. Quello che conta è acquisire una certa conoscenza di questi gioiellini cercando di programmarli in modo da soddisfare le proprie esigenze, a tal proposito ciò che sto descrivendo consentirà di operare ampiamente nella logica combinatoria e sufficientemente nella logica sequenziale. Ciò che affronteremo con le CPLD ha il 100% di portabilità sulle FPGA come la XC3S50-4PQG208C.

MODULAZIONE FSK CON PIC10F

La modulazione FSK (Frequency-Shift Keying) è una delle modulazioni più semplici che possono essere utilizzate sia in ambito hobbistico che in ambito professionale per la trasmissione di un segnale digitale tramite un collegamento analogico.

La Modulazione

In elettronica la modulazione è un processo che varia un attributo di un segnale periodico ad alta frequenza, detto portante, in base al valore del segnale modulante. Il modo in cui si realizza la modulazione è paragonabile al metodo con cui un musicista modula un tono (un onda periodica) da uno strumento musicale, variando volume, fase e frequenza. I tre parametri chiave di un segnale periodico sono infatti quelli appena citati: ampiezza, fase e frequenza. Possono essere modificati in base ad un segnale a bassa frequenza detto segnale modulante.
Il segnale portante per eccellenza è un segnale sinusoidale ad alta frequenza ma anche un segnale rettangolare può essere utilizzato efficacemente.
Un dispositivo che realizza la modulazione di segnale è detto "Modulatore", un dispositivo che realizza il processo inverso, ossia dal segnale modulato ad altea frequenza estrae il segnale modulante a bassa frequenza è detto "Demodulatore".
Questo tipo di circuiti sono ampiamente utilizzati nell'ambito delle telecomunicazioni e sono l'abc per quelli del settore.
L'obbiettivo di una modulazione digitale è di trasferire un flusso di bit attraverso una un canale analogico racchiuso tra una determinata banda passante, per esempio la rete telefonica (300-3400Hz) o una banda radio limitata.
L'obbiettivo di una modulazione digitale in banda base, detta anche codifica di linea o line coding, è di trasferire sempre un flusso di bit digitale ma su un canale in banda base, per esempio attraverso un collegamento tramite bus seriale o tramite una rete locale.

Modulazione FSK

La modulazione FSK ossia Frequency-Shift Keying, punta a modificare la frequenza del segnale portante in base al valore del segnale modulante. In particolare l'applicazione in esame realizza una modulazione FSK compatibile con il protocollo Bell-202. Si tratta di un vecchio protocollo sviluppato dalla Bell System per i modem, con una modulazione FSK del segnale audio (AFSK) codifica e trasferisce i dati alla frequenza di 1200 bit per secondo. Il protocollo Bell-202 AFSK utilizza un tono a 1200Hz per trasmettere un 1 binario e un tono a 2200Hz per uno spazio (zero binario).
Il protocollo Bell-202 ha permesso di sviluppare tecniche come il multi-dropping di più modem slave, connessi ad un unico master attraverso un unico canale di comunicazione, in più è stato la base per la sovrapposizione di segnali ai cavi di alimentazione per distanze superiori agli 80Km in applicazioni marine.

Il circuito

Un modulatore FSK comprende una forma d'onda digitale che oscilla tra due frequenze stabilite in base al livello logico di un segnale digitale. Il circuito in Figura 1

genera un onda sinusoidale campionando continuamente un singolo periodo sinusoidale. L'uscita di IC_{2A} è proporzionale alla corrente attraverso R1, R2 e R3. Queste resistenze sono collegate assieme all'ingresso invertente di IC_{2A} polarizzato a Vcc/2.

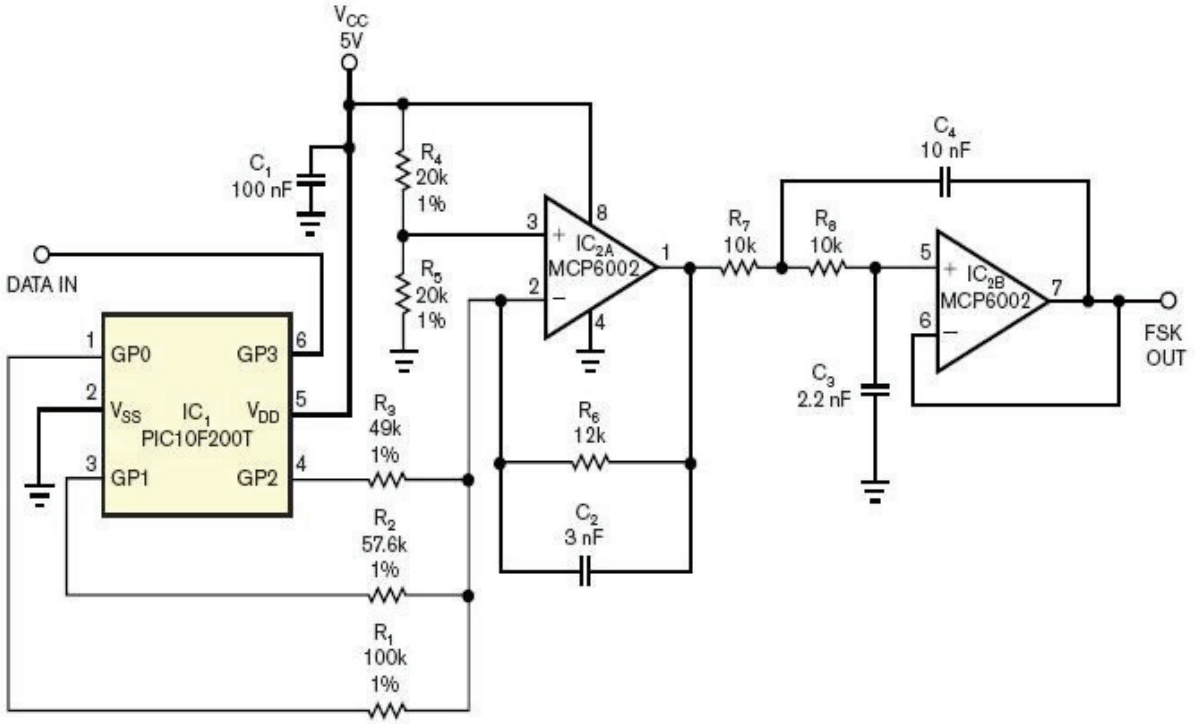


Figura 1: Circuito che realizza una modulazione FSK compatibile con il protocollo Bell-202

Le uscite GP0, GP1 e GP2 del microcontrollore IC_1 producono un treno d'impulsi non sovrapposti. Quando uno dei tre pin genera un segnale alto o basso gli altri sono in una situazione di alta impedenza. Quando uno dei tre pin è a livello alto la tensione ai capi della resistenza collegata è pari a Vcc/2. Quando uno dei tre pin è a livello basso ai capi della resistenza vi è una tensione di –Vcc/2.
Necessario scegliere i valori di R1, R2 e R3 in modo tale che gli impulsi di corrente abbiano ampiezza proporzionale ai campioni della sinusoide a 30°, 60° e 90°, rispettivamente. Quando tutte e tre le uscite del microcontrollore sono in alta impedenza si produce i campione della sinusoide a 0° e non scorre corrente attraverso le resistenze. Quindi partendo da una situazione di alta impedenza per tutte e tre le uscite e in base alla sequenza di livelli alti in uscita su GPO, GP1, GP2 e poi nuovamente GP1 e GP0 si ottiene la generazione del primo semiperiodo positivo dell'onda sinusoidale. Ripetendo il processo ma con le uscite a livello basso si genera il semiperiodo negativo della forma d'onda.
Attraverso questo schema si produce un onda sinusoidale tramite 12 campioni per periodo. In aggiunta rispetto alla frequenza base del'onda generata f_0 si possono inserire componenti a frequenza maggiore pari a $(12k+1)f_0$ e $(12k-1)\ f_0$, con k= 1,2,3 e così via.
La seconda parte del circuito realizza un filtro passa basso, composto da IC_{2B} , R7, R8, C3 e C4 in grado di filtrare le componenti ad alta frequenza ma a bassa ampiezza.

Di seguito il listato in assembly del programma ideato su PIC10F200T. Quando l'ingresso di controllo Data In è a livello alto, la frequenza d'uscita è 1200Hz; quando è a livello basso in uscita si hanno 2200 Hz. Il passaggio da una frequenza all'altra è realizzato in maniera tale da mantenere la continuità di fase. In Figura 2 sono visualizzati il segnale FSK modulato (CH1) e il segnale digitale modulante (CH2).

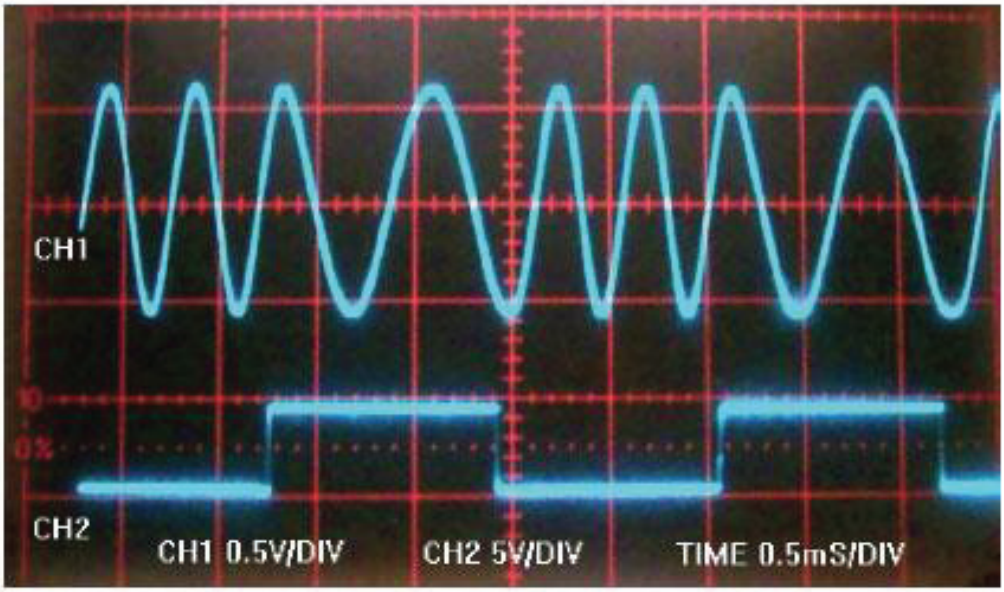


Figura 2: Segnale FSK modulato (CH1) e segnale digitale modulante (CH2)

Listato

```
*************************************************************
* A SIMPLE FSK MODULATOR *
*************************************************************
; THIS CODE IMPLEMENTS A BELL 202 FSK MODULATOR
list p=10F200
include ; This file is part of Microchip MPLAB IDE

;-------------- PIC10F2xx CONFIGURATION -------------
__CONFIG _MCLRE_OFF & _WDT_OFF & _CP_OFF ; External Reset off
; Watch Dog timer off
; Code Protection off

;-------------- VARIABLE DEFINITIONS ----------------
UDATA
PERIOD RES 1 ; Delay counter (Presets to time between sine
samples)
#define DATA_IN GPIO,3 ; GP3 is DATA_IN input pin
OUT_0 EQU b'00001111' ; GP0-GP3 configured as High Impedance
OUT_1 EQU b'00001110' ; Only GP0 configured as output
OUT_2 EQU b'00001101' ; Only GP1 configured as output
OUT_3 EQU b'00001011' ; Only GP2 configured as output
P1 EQU d'19' ; Delay constant for 1200 Hz
P2 EQU d'9' ; Delay constant for 2200 Hz
; Start of program
```

```
RESET CODE 0x000
GOTO init
SUBS CODE 0X10
CHK_DATA: ; Checks Data input pin
MOVLW P1 ; Load W with 1200 Hz Delay constant
BTFSS DATA_IN ; If DATA_IN == High skip next instruction
MOVLW P2 ; Load W with 2200 Hz Delay constant
MOVWF PERIOD ; Load Delay counter;
NOP ; NOP'S added here to fine tune FSK frequencies
NOP
RETLW 0

init:

MOVWF OSCCAL ; Load the factory Internal oscillator calibration
; value
MOVLW OUT_0 ; GP0 - GP3 are OFF (High impedance)
TRIS GPIO
CLRF GPIO ; initialize outputs to Low Level

; initialize OPTION register

MOVLW b'11000111' ; Weak Pull-ups disabled ,Wake-up on pin change
OPTION ; disabled

LOOP: ; This loop produces the twelve sine samples
CALL CHK_DATA ; Check Data
DECF PERIOD,F ; decrement delay counter twice to compensate for
DECF PERIOD,F ; the 6 added instruction cycles:
; GOTO LOOP(2 cycles) DECF PERIOD,F DECF PERIOD,F
; MOVLW 0xFF and MOVWF GPIO
MOVLW OUT_0
TRIS GPIO ; GP0-GP2 are OFF (High impedance)
MOVLW 0xFF
MOVWF GPIO
L1: DECFSZ PERIOD,F
GOTO L1

CALL CHK_DATA
MOVLW OUT_1
TRIS GPIO ; GP0 is High
L2: DECFSZ PERIOD,F
GOTO L2

CALL CHK_DATA
MOVLW OUT_2
TRIS GPIO ; GP1 is High
L3: DECFSZ PERIOD,F
```

```
GOTO L3

CALL CHK_DATA
MOVLW OUT_3
TRIS GPIO ; GP2 is High
L4: DECFSZ PERIOD,F
GOTO L4

CALL CHK_DATA
MOVLW OUT_2
TRIS GPIO ; GP1 is High
L5: DECFSZ PERIOD,F
GOTO L5

CALL CHK_DATA
MOVLW OUT_1
TRIS GPIO ; GP0 is High
L6: DECFSZ PERIOD,F
GOTO L6

CALL CHK_DATA
DECF PERIOD,F ; decrement delay counter to compensate for the
; following 3 added instruction cycles
; (DECF PERIOD,F MOVLW 0x00 and MOVWF GPIO )
MOVLW OUT_0
TRIS GPIO ; GP0-GP2 are OFF (High impedance)
MOVLW 0x00
MOVWF GPIO
L7: DECFSZ PERIOD,F
GOTO L7

CALL CHK_DATA
MOVLW OUT_1
TRIS GPIO ; GP0 is Low
L8: DECFSZ PERIOD,F
GOTO L8

CALL CHK_DATA
MOVLW OUT_2
TRIS GPIO ; GP1 is Low
L9: DECFSZ PERIOD,F
GOTO L9

CALL CHK_DATA
MOVLW OUT_3
TRIS GPIO ; GP2 is Low
L10: DECFSZ PERIOD,F
GOTO L10

CALL CHK_DATA
MOVLW OUT_2
```

```
TRIS GPIO ; GP1 is Low
L11: DECFSZ PERIOD,F
GOTO L11

CALL CHK_DATA
MOVLW OUT_1
TRIS GPIO ; GP0 is Low
L12: DECFSZ PERIOD,F
GOTO L12

GOTO LOOP
END
```

Bibliografia e riferimenti

- http://it.wikipedia.org/wiki/Modulazione
- http://en.wikipedia.org/wiki/Bell_202_modem
- MICROCHIP website: http://www.microchip.com/

“GLUE LOGIC”

Scopo di questo articolo è quello di analizzare l’impiego dei microcontrollori di fascia bassa nella realizzazione di interfacce “glue logic”, ampiamente utilizzate per la connessione tra diversi componenti elettronici ad elevata integrazione.

Il termine “glue logic”, tradotto letteralmente dall’inglese all’italiano, significa “logica collante”, cioè appunto una rete logica, spesse volte ma non necessariamente, di tipo combinatorio che serve ad interfacciare dispositivi elettronici ad elevata integrazione (quali ad esempio CPU, memorie, ADC, DAC, display, ecc.) implementando di volta in volta le funzionalità richieste per attuare la connessione. Le interfacce “glue logic” vengono soventemente indicate nella letteratura tecnica italiana con il termine di “logica sparsa”, proprio perchè a livello pratico vengono realizzate con l’impiego di circuiti logici “sparsi” attorno ai dispositivi più complessi di cui realizzano l’interfacciamento (o se vogliamo, di cui costituiscono il “collante”).

La realizzazione delle interfacce “glue logic”

Le interfacce “glue logic” possono essere realizzate adottando una delle seguenti quattro tecniche:

1 - impiegando logiche standard già progettate per svolgere ben determinate funzioni, come ad esempio i circuiti appartenenti alle note famiglie 7400 (bipolari) oppure 4000 (CMOS). Le logiche di questo tipo sono state storicamente le prime ad essere impiegate, e sono tuttora molto valide per circuiti non particolarmente complessi in quanto offrono dei costi molto contenuti

2 - impiegando le logiche di tipo CPLD (Complex Programmable Logic Device). Le CPLD sono componenti programmabili e riprogrammabili in grado di simulare il comportamento di circuiti logici di complessità non eccessivamente elevata. Presentano l’importante caratteristica di mantenere la programmazione anche a seguito di uno spegnimento, in quanto questa viene conservata in una memoria non volatile. Questa è una significativa differenza rispetto alle FPGA, in cui in genere la programmazione, mantenuta in un file “immagine” su memoria non volatile, deve essere caricata ad ogni accensione o dopo ogni reset hardware. Le CPLD derivano direttamente dalle GAL (Gate Array Logic) molto usate per la realizzazione di interfacce “glue logic” negli anni passati. L’architettura di una CPLD è basata sulle macrocelle, dove per macrocella si intende una logica programmabile usata per generare delle somme di prodotti. Le macrocelle sono poi raggruppate insieme per formare i blocchi funzionali; sia i blocchi funzionali che le macrocelle sono collegati tra di loro tramite interconnessioni programmabili. L’informazione relativa alla configurazioni di queste interconnessioni, come detto, è mantenuta in memoria non volatile.

3 - impiegando le logiche FPGA (Field Programmable Logic Array). E’ stato stimato che almeno la metà dei sistemi embedded siano dotati di un dispositivo FPGA abbinato ad un microcontrollore. Le FPGA possono essere utilizzate per svolgere diversi tipi di funzionalità: dall’implementazione di reti glue logic, all’implementazone di intellectual property (IP), fino ad acceleratori per svolgere algoritmi di elevata complessità. Le FPGA presentano la caratterisitica di essere estremamente flessibili, e, svolgendo molte delle funzionalità di calcolo che normalmente sarebbero a carico del microprocessore, aiutano a migliorare notevolmente le prestazioni del sistema.

Sia le FPGA che le CPLD vengono solitamente utilizzate per realizzare interfacce di complessità medio-elevata. Tuttavia occorre sottolineare come il costo sempre più contenuto, unitamente alla disponibilità di sistemi di sviluppo semplici da usare e molto performanti, stiano imponendo anche questi tipi di logiche nel settore delle interfacce “glue logic” di complessità standard.

Figura 1: la famiglia di CPLD Xilink CoolRunner.

4 - impiegando dei normali microcontrollori di fascia bassa. L’utilizzo di questi tipi di dispositivi in questa nuova veste, per certi versi non convenzionale, è proprio l’oggetto di questo articolo. Le caratteristiche peculiari dei microcontrollori di fascia bassa si possono così sintetizzare:
- dispongono di memorie flash, RAM, ed EEPROM integrate on-chip
- hanno in genere un numero sufficiente di pin configurabili singolarmente volta per volta come ingresso oppure uscita
- offrono la possibilità di essere programmati e riprogrammati, anche con il dispositivo inserito nel circuito senza doverlo rimuovere
- presentano bassi assorbimenti di corrente e possono pertanto essere alimentati con basse tensioni o con semplici batterie

Un primo esempio pratico: il multiplexer da 4 a 1.

Consideriamo come caso pratico quello di un multiplexer da 4 a 1. Un circuito di questo tipo presenta le seguenti caratteristiche:

- - 4 linee di ingresso dati, I0-I3
- - 2 linee di selezione (indirizzamento degli ingressi), S0 ed S1
- - 1 linea di abilitazione del circuito, /E
- - 1 linea di uscita, Y

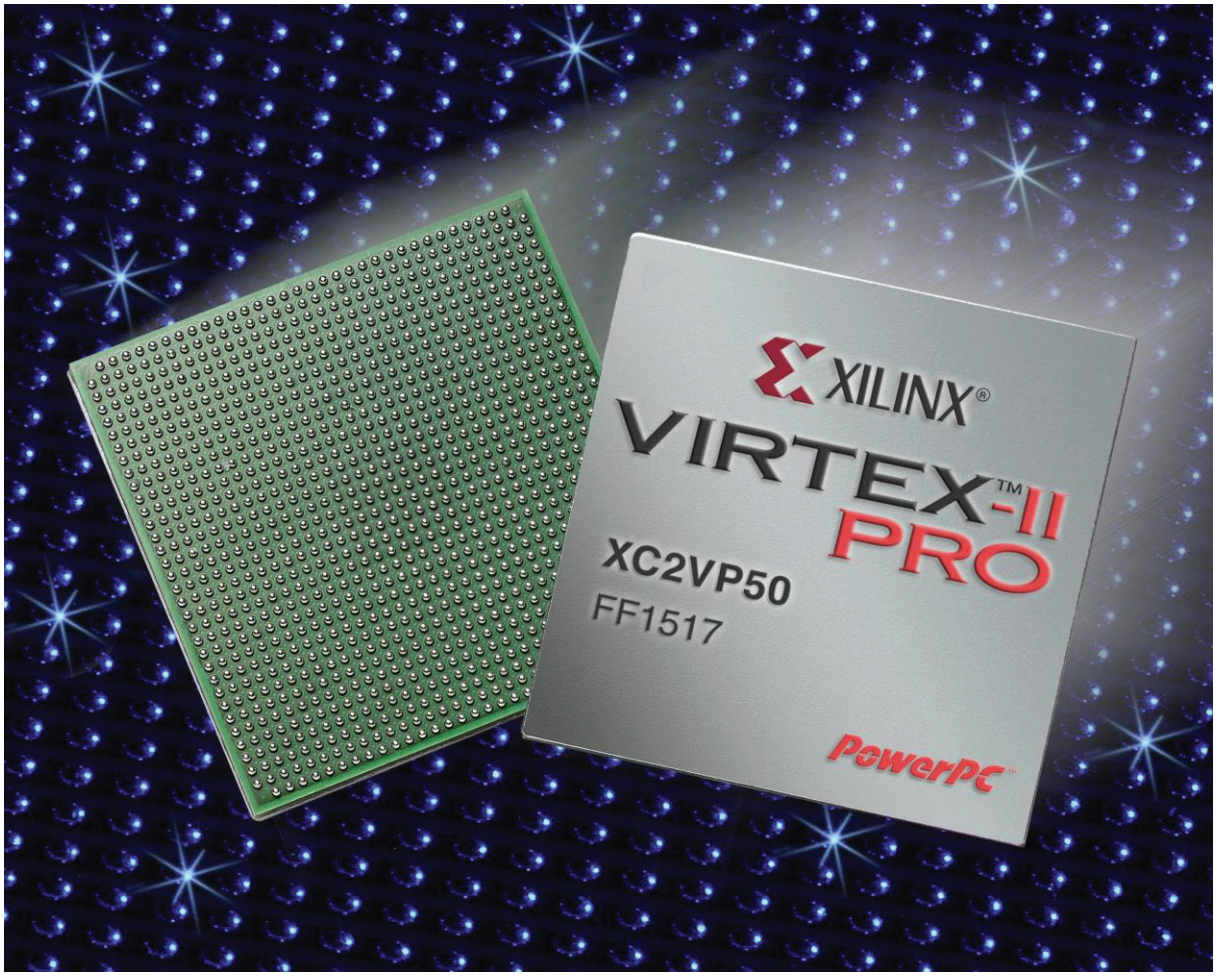


Figura 2: la FPGA Virtex-II Pro di Xilink.

Il comportamento di questo circuito è molto semplice: se la linea di abilitazione si trova al livello logico alto, il circuito multiplexer è inibito e l'uscita si porta sempre al valore logico basso indipendentemente dal valore assunto dagli ingressi; se invece la linea di abilitazione si trova al livello logico basso, l'uscita del circuito assume lo stesso stato logico dell'ingresso indirizzato dalla coppia di linee (bit) di selezione. In figura 3 è mostrato lo schema a blocchi del circuito, mentre nella tabella 1 è riportata la tabella di verità corrispondente al multiplexer 4-1.

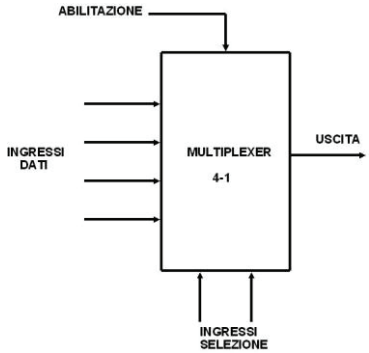


Figura 3: lo schema a blocchi del multiplexer 4-1.

Dall'esame della tabella della verità è possibile ricavare lo schema logico del multiplexer 4-1, cioè la rete combinatoria che lo rappresenta, come visibile in figura 4. A questo punto possiamo decidere come implementare a livello fisico la nostra rete combinatoria, a tutti gli effetti una vera e propria "glue logic". In genere possiamo individuare tre tipi di alternative:

1 - implementare la rete combinatoria tramite componenti discreti reperibili sul mercato. Nel caso si voglia fare ricorso alla ben nota famiglia di integrati digitali 7400, saranno necessari i seguenti componenti: 1 integrato 7404 per realizzare i tre NOT, 1 integrato 7432 per realizzare la porta OR a 4 ingressi (connettendo in cascata 3 porte OR a 2 ingressi), ed infine 2 integrati 7421 per realizzare le 4 porte AND a 4 ingressi.

2 - utilizzare (se esistente) un componente commerciale già disponibile sul mercato ed in grado di svolgere autonomamente la funzionalità richiesta. Nel caso molto semplice preso come esempio, il componente in questione esiste, ed è possibile ad esempio impiegare l'integrato 74153, un doppio selettore/multiplexer da 4 linee ad 1 linea. Lo schema logico mostrato in figura 4 è stato proprio tratto dal datasheet di questo componente (trattasi del 74F153 di Philips).

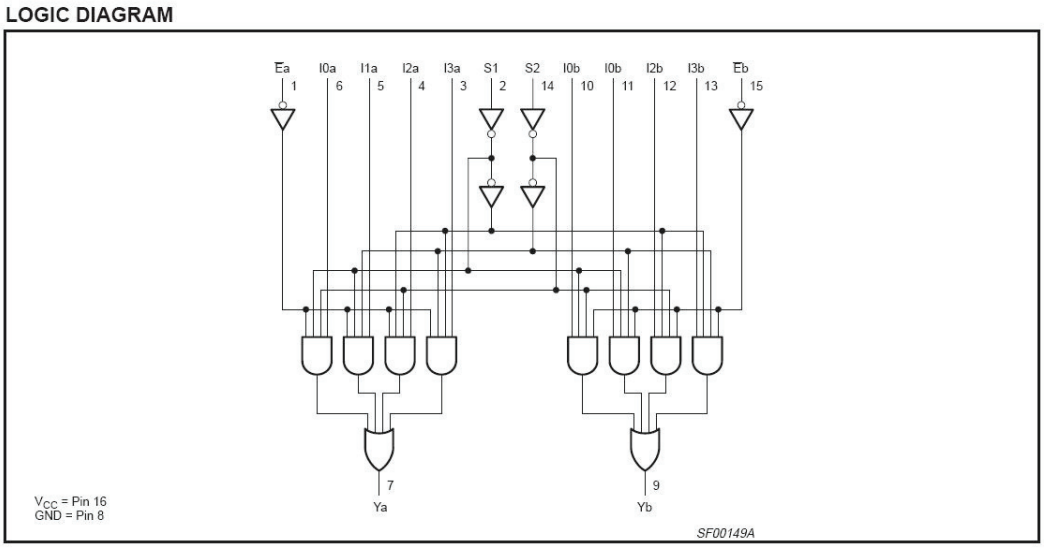


Figura 4: lo schema logico del multiplexer 4-1.

L'integrato dispone complessivamente di 16 pin, e ciascun circuito multiplexer dispone di ingressi, uscita, e segnale di strobe (abilitazione) separati, mentre le due linee di selezione sono condivise tra i due circuiti.

3 - realizzare la glue logic impiegando un microcontrollore di fascia bassa. Sul mercato esiste una moltitudine di MCU candidate a svolgere questo ruolo; nel nostro caso ci soffermiamo sulla famiglia dei microcontrollori PIC di Microchip essenzialmente per i seguenti motivi: sono facilmente reperibili sul mercato ad un prezzo contenuto, sono programmabili attraverso numerosi sistemi di sviluppo che includono oltre all'assembler anche linguaggi di alto livello come il C ed il Basic, ed infine sono molto diffusi non soltanto negli ambiti specialistici, ma anche tra gli hobbysti ed i progettisti non professionali. Il listato 1 mostra quello che potrebbe essere il codice in grado di svolgere la funzionalità richiesta su un PIC 16F84A, compilato con il compilatore CC5X di B Knudsen Data (un qualunque altro compilatore C per PIC va benissimo). Tutti i segnali di I/O sono stati mappati sulla porta B (B0-B6 input, B7 output). Tra le principali applicazioni dei circuiti multiplexer ricordiamo la possibilità di selezionare uno ed un solo ingresso tra N linee disponibili. Questa applicazione può ad esempio essere impiegata per eseguire la scansione selettiva delle linee associate ad una matrice di tasti. Un'altra possibilità di eseguire la conversione da parallelo a seriale. Attivando l'ingresso di abilitazione (strobe) tramite un segnale di clock, si ottiene una conversione parallelo/seriale per parole composte da un solo nibble. Esistono sul mercato multiplexer anche per 8 ingressi, per cui diventa molto semplice realizzare un circuito che esegua la conversione parallelo/seriale, alla velocità definita dalla frequenza del clock che comanda il segnale di strobe, per parole da 1 byte.

Un secondo esempio pratico: priority encoder da 8 a 3 linee

L'encoder (detto anche demultiplexer) con priorità che andremo a considerare dispone di 8 linee in ingresso e 3 linee in uscita. Il funzionamento è analogo a quello dei normali circuiti demultiplexer, in cui l'uscita viene selezionata in base alla codifica binaria presente in ingresso, con la seguente importante differenza: l'uscita selezionata è sempre quella corrispondente all'input attivo di più alta priorità (la priorità è in ordine decrescente procedendo dal bit di ingresso 7 verso il bit 0). In questo caso facciamo subito riferimento all'integrato 74F148, in grado di svolgere la funzionalità in esame. Questo componente dispone di un'ulteriore ingresso, /EI, che serve ad abilitare il circuito, e di due ulteriori uscite, /GS ed /EO: /GS è attivo basso quando uno qualunque degli ingressi è basso (Group Signal Otput), mentre /EO è attivo basso quando tutti gli ingressi sono alti. La tabella della verità corrispondente è riportata in tabella 2. In Figura 5 è invece riportato lo schema logico del circuito, come indicato sul datashet del componente.

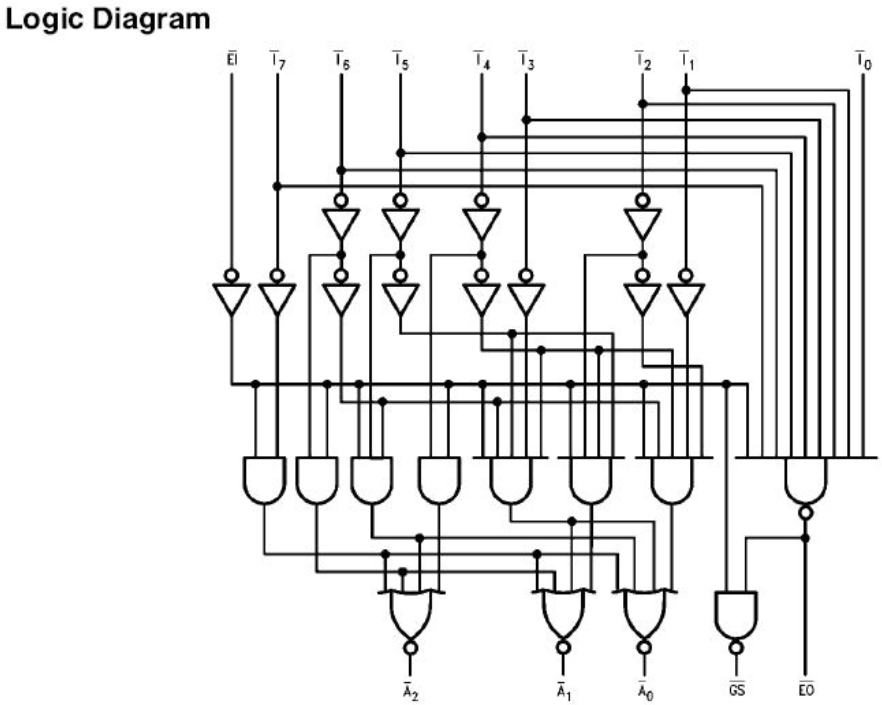


Figura 5: lo schema logico dell'encoder priority 8-3.

Per realizzare il comportamento di questo circuito tramite il PIC 16F84A, che dispone complessivamente di 13 pin di I/O, possiamo pensare di rinunciare al segnale di output /EO, pealtro significativo quando più encoder vengano connessi in cascata. Possiamo pertanto pensare di mappare gli 8 ingressi dati tutti sulla porta B, mentre alla porta A faranno capo l'ingresso di abilitazione, ed i segnali di output A0-A2 e GS. Il codice corrispondente è riportato nel listato2. Ricordiamo che i priority encoder vengono utilizzati tipicamente nelle applicazioni in cui diversi dispositivi (memorie, CPU, dispositivi di I/O) debbano condividere una risorsa comune, per esempio il bus. Ad ogni dispositivo viene pertanto assegnata una priorità (in funzione ad esempio della sua posizione sul bus) in modo tale che in caso di conflitto venga privilegiato il dispositivo con priorità maggiore.

Conclusioni

Abbiamo visto in questo articolo come anche i microcontrollori più semplici possano essere utilizzati per realizzare dei circuiti glue logic di tipo programmabile e a basso costo, senza ricorrere all'utilizzo di componenti quali CPLD ed FPGA, più adatti ad applicazioni in cui l'elevata velocità di calcolo e/o la complessità funzionale siano requisiti fondamentali.

Ricevitore per la banda CB

Se avete intenzione di muovere i primi passi nel modo della radiotecnica e magari avete l'ambizione di diventare radioamatore, ecco qui un "circuitino" da costruire con poca spesa e in poco tempo che vi darà la possibilità di ascoltare le comunicazioni in modulazione di ampiezza tra gli operatori della banda CB.

Certo in un'epoca in cui la telefonia cellulare e internet la fanno da padroni, parlare di radiotecnica può sembrare anacronistico; ma questa disciplina è ancora alla base delle moderne radio comunicazioni.
Veniamo al nostro ricevitore; la gamma di frequenze scelte è quella delle onde corte più precisamente dei 27 Mhz (da 26,965 a 27,405 Mhz) che il mistero delle telecomunicazioni ha assegnato all'uso privato, da qui il nome di banda CB (Citizen Band), tradotto Banda Cittadina. Per operare è necessario acquistare un'apparto omologato e fare una semplice dichiarazione di inizio attività pagando una tassa annuale per l'esercizio della stazione. Su queste frequenze è possibile trasmettere in modulazione d'ampiezza (AM), di frequenza (FM) e in banda laterale singola (SSB) con potenze non superiori ai 4 Watt. Gli operatori CB, come tutti gli utenti di sistemi di comunicazione deregolamentati, non vanno assolutamente confusi con i radioamatori, che invece devono superare un esame scritto per essere autorizzati a trasmettere e che possono utilizzare una gamma di frequenze radio più vasta con apparecchi di potenza decisamente più elevata. Chi desidera diventare radioamatore e ha intenzione di sostenere il relativo esame, può benissimo iniziare ad esercitarsi sulle frequenze della CB cercando di prendere dimestichezza con tutte quelle che sono le caratteristiche di una stazione radiotrasmittente. In questo contesto il ricevitore che vi vado a proporre può essere utile per capire come funzionano i collegamenti radio in questa banda, quale è il gergo usato chi sono gli operatori CB che vi stanno più vicino e quanto altro scoprirete ascoltando.

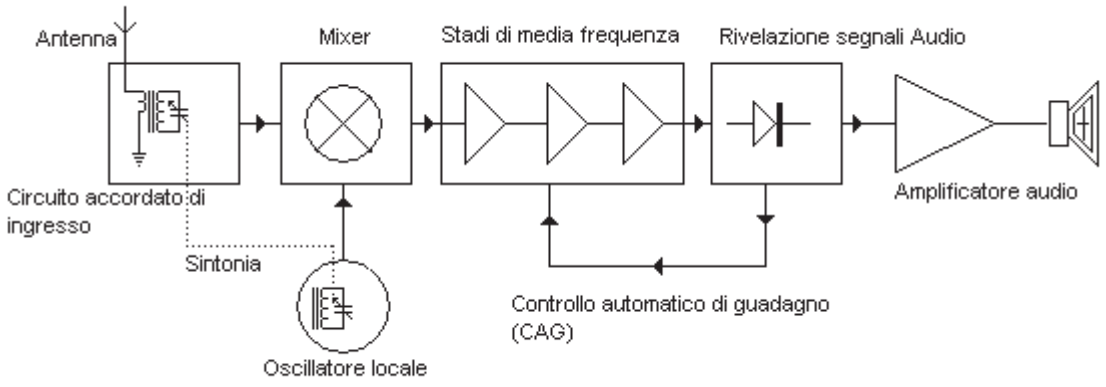


Figura 1: schema a blocchi di una supereterodina.

La supereterodina

La configurazione scelta per il nostro ricevitore è una classica supereterodina a singola conversione, che rappresenta un buon compromesso tra semplicità

costruttiva e prestazioni. Ma vediamo come funziona un ricevitore di questo tipo. Il principio è molto semplice, volendo realizzare un ricevitore capace di sintonizzare più di una stazione occorrerebbe, in linea di principio, predisporre un sistema di filtri capaci di selezionare e successivamente amplificare e rilevare ognuna delle stazioni che si intende ascoltare. Questo meccanismo anche se concettualmente corretto, oltre ad essere antieconomico sarebbe comunque molto critico sia da attuare che e da mettere a punto. Questi motivi portarono, agli inizi del secolo scorso, allo sviluppo di un sistema capace di convertire tutti i segnali radio ricevibili in una determinata banda frequenze ad una frequenza fissa che in seguito prese il nome di frequenza intermedia e fu così che nacque il ricevitore supereterodina. Dallo schema a blocchi di figura 1 è possibile capire il funzionamento di un ricevitore di questo tipo. In pratica il sistema che rende possibile la trasformazione del segnale ricevuto ad uno di frequenza fissa è il Mixer, si tratta di una configurazione circuitale in cui sono miscelati assieme il segnale radio ricevuto in antenna con quello generato da un oscillatore locale, il battimento di tali segnali produce a sua volta due nuovi segnali aventi rispettivamente una frequenza pari alla somma e alla differenza delle frequenze dei due segnali miscelati. A questo punto per estrarre la nostra frequenza intermedia si pone in uscita al Mixer un circuito accordato che filtri il segnale con frequenza più bassa e quindi meno critico da amplificare, che nel nostro caso sarà quello con frequenza pari alla differenza delle frequenze due segnali miscelati. Al fine di ottenere una frequenza intermedia fissa è necessario che il circuito accordato d'ingresso e quello dell'oscillatore locale siano sintonizzati, di volta in volta, sempre in maniera tale che la differenza di frequenza sia pari al valore della frequenza intermedia e questo si ottiene con la taratura iniziale del ricevitore, dopo di che sarà sufficiente sincronizzare la frequenza di risonanza dei due circuiti utilizzando ad esempio il consueto condensatore variabile doppio o come nel nostro caso due diodi varicap uno per il circuito d'ingresso e uno per l'oscillatore locale la cui polarizzazione è governata contemporaneamente dallo stesso potenziometro. Gli stadi successivi al Mixer hanno il solo scopo di amplificare il segnale a frequenza intermedia e sono chiamati stadi di media frequenza, dopo di loro, c'è il rivelatore audio che nel caso di ricevitori per segnali modulati in ampiezza è spesso costituito da un semplice diodo al germanio. Per evitare che segnali molto forti saturino gli amplificatori di media frequenza, rendendo non più intelligibile il segnale ricevuto, il guadagno di quest'ultimi è regolato da una parte del segnale audio retrocesso (CAG); il tutto alla fine è completato da un amplificatore a bassa frequenza che permette l'ascolto in cuffia o in altoparlante.

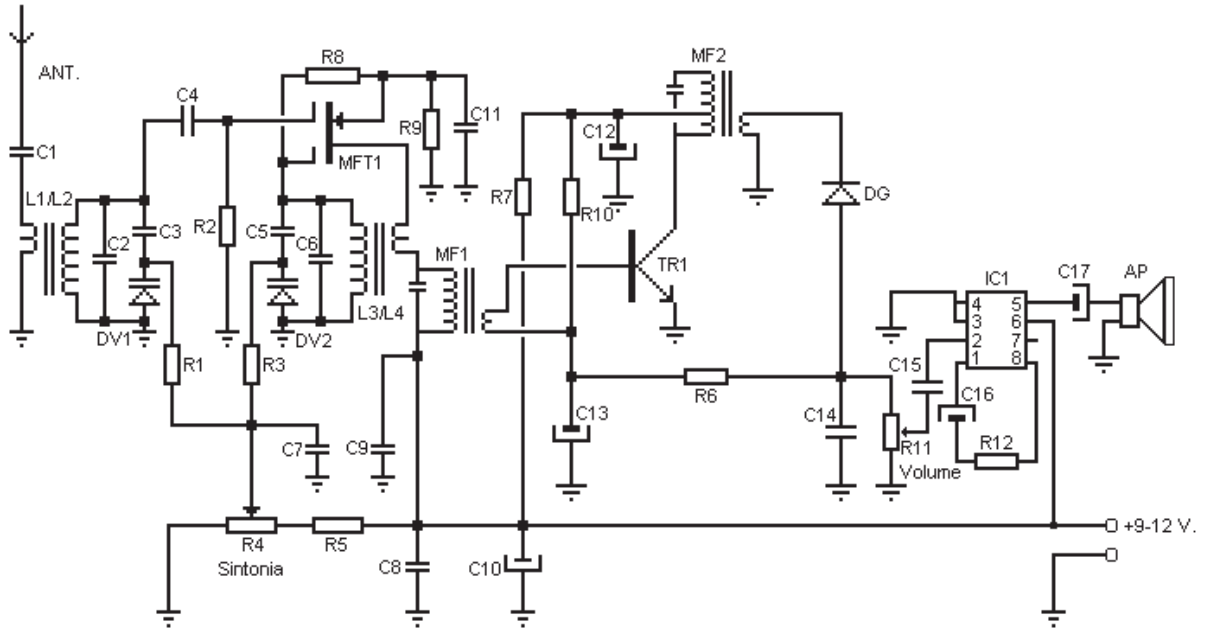


Figura 2: schema elettrico del ricevitore.

Schema Elettrico

In realtà la nostra supereterodina è molto semplificata, le funzioni di Mixer e oscillatore locale sono svolte da un unico componente, ovvero il mosfet MFT1. Come si può vedere dallo schema elettrico (figura 2), il segnale ricevuto dall'antenna e sintonizzato tramite il gruppo LC composto da L2, C2, DV1 e componenti limitrofi, viene applicato tramite C4 al gate 1 del nostro mosfet; contemporaneamente l'accoppiamento induttivo tra L3 e L4 genera una reazione positiva tra il gate 2 e il drain tale da portare il mosfet ad oscillare alla frequenza determinata dal circuito risonante realizzato con la bobina L3 da C6 e dal diodo varicap DV2.

A questo punto sul drain del nostro mosfet è possibile selezionare tramite MF1 il segnale ottenuto per battimento tra quello generato localmente e quello ricevuto in antenna. Selezionato il segnale a frequenza intermedia, che nel nostro caso è di 455 Khz, sarà compito di TR1 amplificarlo ad un livello tale da renderlo facilmente rivelabile. In pratica, anche qui per semplicità, abbiamo ridotto "all'osso" la catena di amplificazione di media frequenza realizzandola con un solo stadio quello, appunto, pilotato dal transistor TR1. All'uscita del secondo e ultimo trasformatore di media frequenza troviamo il diodo al germanio, che come detto sopra, ha il compito di rivelare la componente in bassa frequenza del segnale radio e inviarla allo stadio amplificatore audio (LM386) per essere riprodotta in altoparlante. Da notare che il diodo è inserito in maniera tale da rilevare le semionde negative del segnale, in questo modo tramite la resistenza R6 e il condensatore C13, è possibile gestire il punto di lavoro di TR1 in funzione dell'ampiezza del segnale audio rivelato realizzando così un efficace controllo automatico di guadagno.

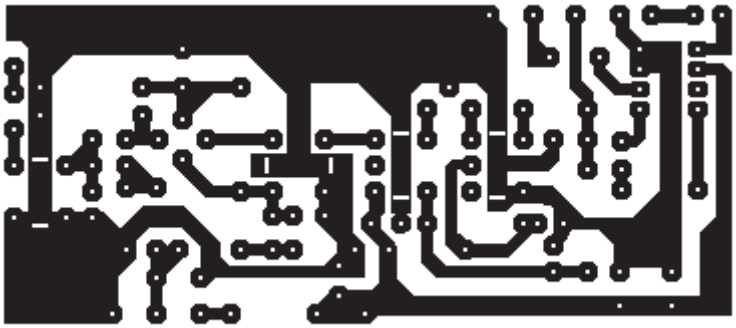


Figura 3: il circuito stampato.

Montaggio e taratura

Visto che si tratta di un circuito in alta frequenza per il montaggio del nostro ricevitore conviene avvalersi di un apposito circuito stampato (figura 3). I componenti utilizzati non sono critici, sia il mosfet che il transistor possono essere sostituti con degli equivalenti. Per i varicap ho usato dei BB139, ma anche qui possiamo utilizzare dei sostituti, l'importante è che la capacità abbia un'escursione compresa tra pochi PF ad un massimo di 30-40 PF. Le medie frequenze sono quelle a 455 Khz, 10x10 mm, utilizzate nei vecchi ricevitori per onde medie; la prima MF1 deve essere quella con nucleo GIALLO (AM1) e la seconda MF2 è quella con nucleo NERO(AM3).

Le bobine

Un discorso a parte va fatto per le bobine L1/L2 e L3/L4 le quali vanno avvolte a mano utilizzando un supporto con nucleo in ferrite regolabile, quelli che ho utilizzato per il prototipo, vedi figura 4, hanno una base 10x10 mm e nucleo da 5 mm.

- La bobina L1/L2 va realizzata partendo da L2 avvolgendo 15 spire di filo di rame smaltato da 0.4-0.5 mm iniziando dal lato collegato a C3/C4. L1 è un semplice link 2 o 3 spire fatto con del filo per collegamenti uni conduttore isolato in plastica da 0.6 mm avvolto direttamente sopra a L2.
- L3/L4 va avvolta iniziando da L3 che come L2 dovrà essere realizzata avvolgendo 15 spire di filo di rame smaltato da 0,4-0,5 mm iniziando dal lato collegato al gate 2 del mosfet.

Il link L4 dovrà essere avvolto di seguito a L2 utilizzando lo stesso filo di rame smaltato partendo dal lato collegato alla media frequenza MF1. Queste due bobine vanno avvolte con cura, in particolare fate attenzione a come avvolgete il link L4 della bobina L3/L4, se invertite gli estremi di questo link il mosfet non entra in oscillazione e il ricevitore resta ovviamente muto, sempre per L3/L4 è necessario, in fase di montaggio racchiuderla in apposito schermo metallico che dovrà essere saldato a massa. Per chi è in vena di sperimentazioni può provare a modificare L2 e L3 aumentando o diminuendo leggermente il numero spire e/o modificando il valore dei condensatori C2 e C6; in questo modo sarà possibile ricevere altre gamme delle onde corte.

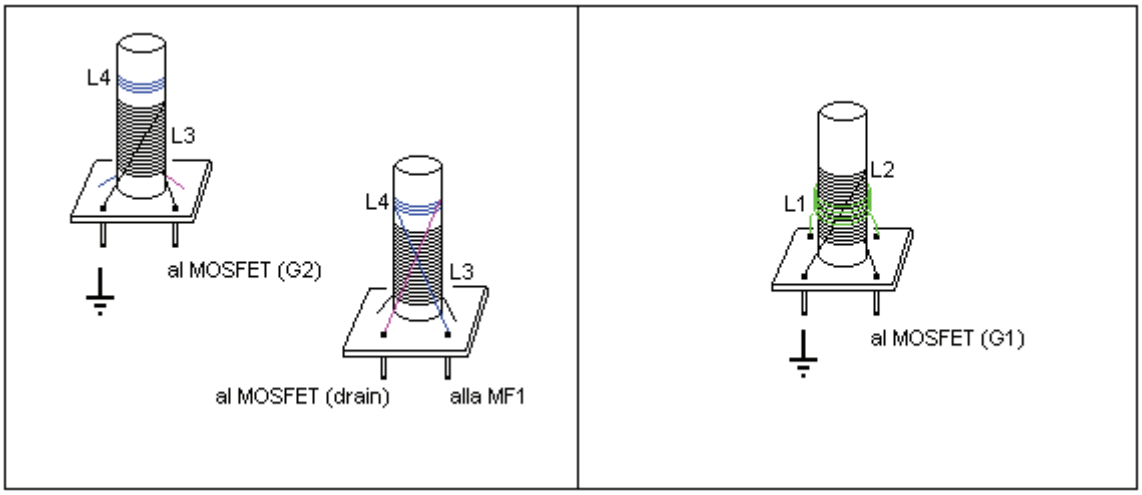


Figura 4: realizzazione delle bobine.

Il piano di montaggio

In figura 5, potete vedere come sono disposti i componenti sul circuito stampato, partite con il montare quelli di profilo più basso ed inserite per ultimo le bobine e le medie frequenze. Come di consueto controllate il corretto inserimento dei semiconduttori diodi, transistor e mosfet, mentre per l'integrato vi consiglio di montarlo su uno zoccolo per circuiti integrati.

Taratura

Il ricevitore per poter funzionare necessita ovviamente di una taratura la quale consisterà sostanzialmente nella messa in gamma delle bobine L1/L2 e L3/L4 e nell'allineamento delle medie frequenze. Per ottenere una perfetta taratura sarebbe necessario possedere almeno un frequenzimetro e un generatore RF modulato in ampiezza, chi possiede questi strumenti sa sicuramente come utilizzarli, io preferisco descrivervi un metodo più empirico adatto anche a chi ancora non ha un laboratorio molto attrezzato. Cominciamo con il ruotare il nucleo di MF1 e MF2 a metà corsa, stessa cosa fate con i nuclei delle bobine L1/L2, L3/L4 e del potenziometro di sintonia R4. Date alimentazione al circuito e collegate uno spezzone di filo lungo un paio di metri come antenna, agendo sul volume dovreste sentire in altoparlante il consueto rumore di fondo, non avendo un generatore RF, dovrete andare a "caccia" di un segnale radio da utilizzare come riferimento per fare questo dovrete agire sul nucleo della bobina L3/L4 (oscillatore locale) e sulla sintonia, potenziometro R4, fino a captare qualche emittente CB. Trovato un segnale radio il passo successivo è quello di ruotare L1/L2 per rendere massimo il livello del segnale audio in altoparlante, successivamente agite sui nuclei delle medie frequenze partendo prima da MF2 e passando successivamente su MF1 con l'obiettivo sempre di massimizzare il segnale ricevuto. Dopo un po' di tentativi vedrete che riuscirete a ricevere senza difficoltà i CB più vicini alla vostra abitazione. Vi ricordo comunque che questo piccolo ricevitore per quanto ben tarato non può competere in termini di sensibilità e selettività con quelli normalmente presenti negli apparati commerciali, ma nessuno vi potrà togliere la soddisfazione di poter ascoltare quello che succede su queste frequenze con un ricevitore realizzato completamente da voi stessi.

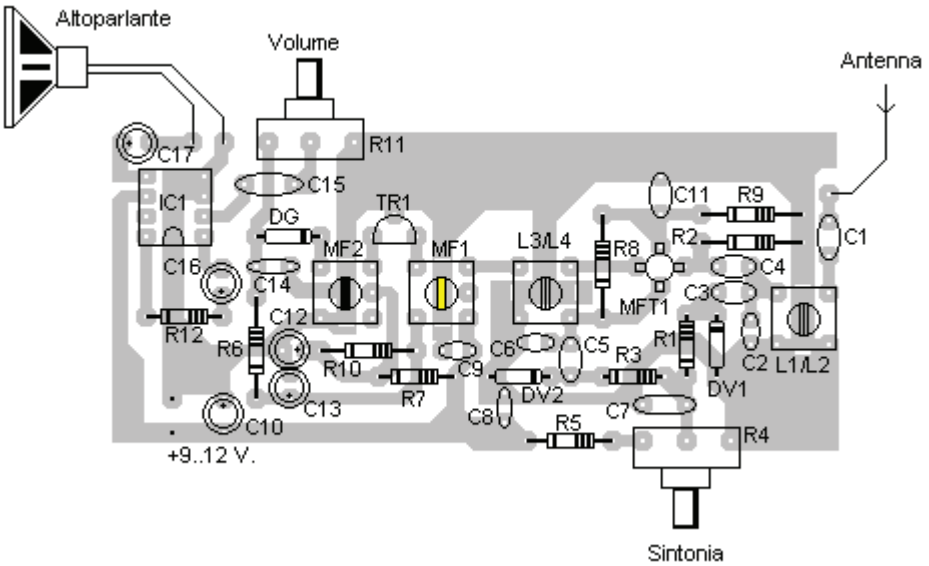


Figura 5: piano di montaggio.

Lista componenti

R1	39 Kohm ¼ w	C8	100 nf 63V. Cer.
R2	100 Kohm ¼ w.	C9	100 nf 63V. Cer.
R3	39 Kohm ¼ w	C10	47 µf 25V. Elettr
R4	47 Kohm pot. Lin.	C11	10 nf 63V. Cer.
R5	390 ohm ¼ w	C12	10 µf 25V. Elettr.
R6	22 Kohm ¼ w	C13	10 µf 25V. Elettr.
R7	220 ohm ¼ w	C14	4700 pf 63 V. Cer.
R8	1 Mohm ¼ w	C15	680 nf poliestere
R9	220 ohm ¼ w	C16	10 µf 25V. Elettr.
R10	56 Kohm ¼ w	C17	100 µf 25V. Elettr.
R11	4,7 Kohm pot. Log.	MF1	AM1 nucleo giallo
R12	1 Kohm ¼ w	MF2	AM3 nucleo nero
C1	22 pf 63V. Cer.	L1/L2/L3/L4	Vedi testo
C2	39 pf 63V. Cer.	IC1	LM 386
C3	150 pf 63V. Cer.	MFT1	BF960
C4	22 pf 63V. Cer.	TR1	BF199
C5	150 pf 63V. Cer.	DV1	BB139
C6	39 pf 63V. Cer.	DV2	BB139
C7	100 nf 63V. Cer.	DG	AA137

Microchip presenta la prima Piattaforma di Sviluppo con 2D Multi-touch e 3D Gestures

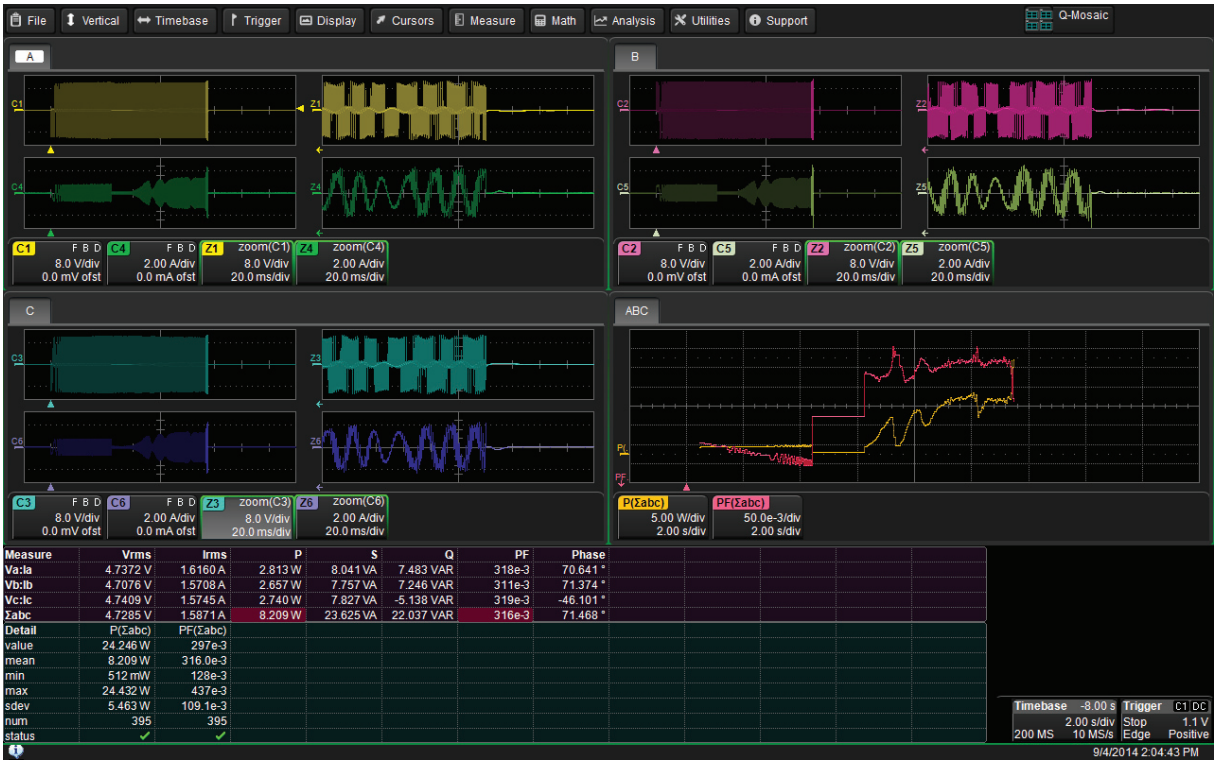
Microchip annuncia l'ampliamento del suo portfolio di Human Interface Input Sensing Solution con il nuovo 3DTouchPad, una periferica PC e prima piattaforma di sviluppo al mondo per 2D multi-touch e 3D gestures.

Il 3DTouchPad reinventa l'input sensing aggiungendo il riconoscimento gestuale in spazio libero al multi-touch capacitivo proiettato quale solida e conveniente piattaforma di sviluppo e progetto di riferimento. Il 3DTouchPad è la prima piattaforma di sviluppo al mondo a combinare input 2D multi-touch e 3D air gesture focalizzata sul mercato delle periferiche PC. Il 3DTouchPad offre un solito ed innovativo riconoscimento gestuale 3D utilizzando la tecnologia GestIC® di Microchip che offre un range di rilevamento fino a 10cm per 3D gestures, oltre alla soluzione Microchip altamente sensibile di 2D multi-touch capacitivo proiettato in grado di supportare fino a 10 punti di tatto e gesture superficiale multi-dita. Il 2D multi-touch è ottimizzato dalla nuova linea Microchip di driver per touch-screen capacitivo, MTCH652, che è stato anch'esso annunciato oggi.
Il nuovo 3DTouchPad include funzionalità driverless e out-of-the-box per Windows® 7/8.X e OS X®, 3D air gestures, prestazioni multi-touch avanzate che comprendono gesture superficiale, e un GUI e SDK/API package scaricabile gratuitamente e appositamente realizzato per gli sviluppatori. Le possibilità per questa tecnologia si espandono anche oltre il mercato dei PC market e touch pad per computer e include prodotti sanitari hands-free, domotica, controlli remoti, game controller, dispositivi indossabili e applicazioni automotive.

Il kit di sviluppo production-ready include GUI e SDK gratuiti che consentono l'utilizzo di dispositivi di input 2D e 3D innovativi per progetti di interfacce utente avanzate.

Teledyne LeCroy presenta il nuovo software dedicato alla gestione motori "Motor Drive Power Analyzer" per gli oscilloscopi della serie HDO8000.

La nuova funzionalità per l'analisi di potenza nell'ambito della gestione/controllo dei motori trifase utilizza l'oscilloscopio ad alta definizione ad 8 canali analogici e 12 bit di risoluzione.



Teledyne LeCroy annucia il software dedicato all'analisi di potenza e controllo dei motori trifase per i suoi oscilloscopi della serie HDO8000. Gli oscilloscopi a canali misti serie HDO8000, con 8 canali analogici di ingresso, 12 bit di risoluzione e fino ad 1 Ghz di banda passante, sono la soluzione perfetta per il controllo integrato e per il debug della parte di potenza del motor drive. Con l'introduzione della capacità di analisi della potenza trifase integrata con l'analisi della velocità e della coppia del motore, è ora possibile validare e debuggare più velocemente e facilmente un sistema completo di controllo. E' possibile effettuare estensive misure di efficenza e di potenza sia sulla parte di controllo che sul motore.

L'HDO8000 ed il software di analisi di potenza per il motor drive saranno utilizzati dai progettisti ed i produttori di motori, di sistemi di controllo motori, di azionatori a frequenza variabile, di azionatori a velocità variabile e dai produttori di attrezzatura per l'automazione industriale ed il controllo della trazione. I progettisti che integrano i motori nei loro progetti (veicoli, utensili elettrici, elettrodomestici, ascensori, ventole, soffiatori, compressori, pompe, ecc.) utilizzeranno questo software per validare i controlli proprietari ed il progetto del sistema di controllo completo.

Capacità di validazione e di debug del sistema di controllo completo

Un sistema di controllo completo è un insieme complesso di elettronica di potenza trifase, sensori motore/meccanici analogici e digitali, e di controlli integrati con una gran varietà di segnali complessi analogici, digitali, seriali e modulati PWM. Gli oscilloscopi sono in gradi di catturare sistemi di controllo integrati ad alta velocità o attività di transistor di potenza, ma finora erano incapaci di misurare le efficenze e la potenza dei drive. I Power Analyzer sono strumenti ad utilizzo univoco per misurare le efficenze e la potenza dell' ingresso/uscita di un "black box", ma offrono una cattura limitata se non nulla delle forme d'onda per il debug del controllo integrato o del sistema di controllo.

L'oscilloscopio HDO8000 con il software "Motor Drive Power Analyzer" permette l'acquisizione delle forme d'onda dalla sezione di potenza del drive, dai singoli transistors di potenza e dal sistema di controllo integrato, e permette l' analisi di potenza trifase simultanea delle forme d'onda in uno strumento ad alte prestazioni. E così oggi possibile l' analisi e ed il debug completo di tutti gli aspetti del motor drive.

Capacità di setup flessibile

Una interfaccia utente intuitiva e grafica fornisce la guida al setup ed alle connessioni. La capacità di setup è proposta per qualsiasi combinazione di ingresso/uscita del drive sia a singola fase che trifase e supporta entrambi i metodi di calcolo 2 e 3-wattmeter ed una conversione di tensione da line-line a line-neutral. Utilizzando il metodo 2-wattmeter, l'efficenza dell'ingresso/uscita del drive e dell'uscita del motore possono essere misurati utilizzando gli 8 ingressi dell'oscilloscopio HDO8000.

L'integrazione più completa per il motor drive

Il software di analisi di potenza del motor drive permette la semplice integrazione di praticamente qualsiasi tipo di sensore di velocità, rotazione o posizione, includendo tachimetri analodici e digitali (impulsi), Brushless DC (BLDC) Hall sensors, quadrature Encoder Interface (QEI) e Resolvers. Inoltre, i segnali dei sensori di Hall e dei QEI possono essere integrati tramite i canali digitali, preservando i canali di ingresso analogici per altri segnali.

Oltre I "numeri" – Maggiori informazioni

Diversi parametri di tensione, di corrente, di potenza (reale, apparente, reattiva), di angolo di fase, di fattore di potenza e di efficenza sono calcolati sulle forme d'onda di tensione e corrente acquisite e vengono visualizzati su una tabella. La tabella è configurabile dall' utente ed è visualizzata assieme alla forma d' onda acquisita, fornendo le informazioni solitamente fornite da un Power Analyzer.

Inoltre, una forma d' onda che visualizza qualsiasi varazione del parametro di misura su ogni ciclo può essere visualizzata semplicemente selezionando un valore nella tabella. Questa forma d' onda è correlata temporalmente con altre forme d' onda acquisite dall' oscilloscopio HDO8000 e può essere utilizzata per correlare comportamenti complessi del drive ad altre forme d' onda di controllo o del sistema di potenza e per debuggare problemi del sistema di controllo. Possono essere visualizzati anche dettagli statistici delle misure. Questa informazione aggiuntiva va ben oltre le possibilità offerte da un Power Analyzer.

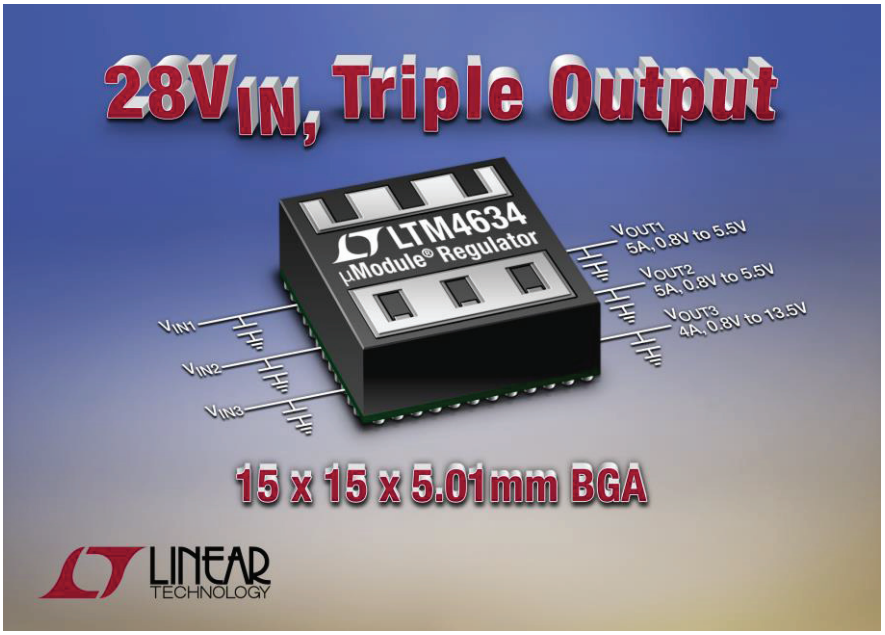
Analisi della risposta dinamica del drive

La lunga memoria di acquisizione dell' HDO8000 (250 Mpunti/ch) fornisce capacità uniche di analisi della risposta dinamica del motore e del drive. Per esempio, si possono catturare 25 secondi di acquisizione continua ad un campionamento di 10Ms/s. Questo permette di capire i comportamenti dinamici del drive, come lo startup, l' applicazione del carico o di condizioni di carico che cambiano velocemente, e di correlare problemi di risposta del drive a errori nelle istruzioni del sistema di controllo o nella sezione di potenza.

Sonde

Le classiche sonde di tensione passive o le sonde di tensione attive e le sonde di corrente possono essere utilizzate per le misure di potenza. In molti casi, le sonde richieste per il debug del controllo integrato o della sezione di potenza possono essere utilizzate per le misure di potenza, rendendo non necessario fare ulteriori investimenti. Altri apparecchi di misura di tensione e di corrente (trasformatori di potenziale o di corrente, trasduttori di corrente, sonde Rogowski, ecc.) possono essere facilmente integrate nell' oscilloscopio HDO8000.
Un isolamento di 1000Vrms è disponibile con le sonde ad alta tensione Teledyne LeCroy della serie HVD con una tensione differenziale massima di 2000Vpk-pk e l'1% di accuratezza. Queste sonde hanno prestazioni eccellenti ad un prezzo molto conveniente.

Regolatore step-down µModule a tre uscite da 28VIN con dissipatore integrato fornisce 70W in 4,5cm^2



Linear Technology Corporation presenta l'LTM4634, un regolatore step-down µModule® (micromodule) a tre uscite in un package BGA di 15 x 15 x 5,01mm con dissipatore di calore integrato. Questo package con funzionalità termiche avanzate consente all'LTM4634 di fornire massima corrente a tutte e tre le uscite (5A, 5A e 4A), più di 70W di potenza di uscita a una temperatura ambiente di 65°C con flusso d'aria di 200LFM. Combinato con solo 13 condensatori ceramici e tre resistenze, l'LTM4634 è adatto a schede a circuiti stampati a doppio lato da 4,5cm2. L'LTM4634 include tre controller DC/DC, switch di potenza, induttori e compensazione in un unico package. Latensione di uscita regolata per i canali 1 e 2 è regolabile da 0,8V a 5,5V e da 0,8V a 13,5V per il terzo canale; tutti e tre i canali hanno una precisione di ±1,5% rispetto a linea, carico e temperatura. La combinazione di ampio range di tensioni di uscita, package ad efficienza termica, regolazione precisa e ingombro ridotto rendono l'LTM4634 adatto a regolare bus di potenza intermedi comuni da 12V, 5V e 3,3V nella strumentazione e in applicazioni di comunicazione e industriali.I tre switch dell'LTM4634 funzionano con una tensione di ingresso compresa tra 4,75V e 28V. Pin di potenza separati per ciascuno switch consentono di alimentare il canale 1 e/o 2 da un bus intermedio da 12,5V, 50W dal canale 3. I tre switch, che funzionano con lo stesso segnale di clock interno o esterno opzionale, hanno uno sfasamento di fase interno di 120° per ridurre al minimo il ripple in ingresso. Per carichi superiori a 5A, i canali 1 e 2 possono essere collegati in parallelo per supportare fino a 10A. Il regolatore include la protezione dei guasti dovuti a sovratensione e sottocorrente delle uscite. La temperatura internadell'LTM4634 può essere osservata tramite due monitor della temperatura interna con diodi posizionati accanto alle fasi di potenza.

L'LTM4634 è già disponibile nella versione con temperatura operativa interna compresa tra –40°C e +125°C. Il prezzo parte da $38,11/cad. per 1.000 pezzi. Per maggiori informazioni, visitare la pagina www.linear.com/product/LTM4634

Riepilogo delle caratteristiche: LTM4634

- Soluzione step-down a tre uscite, 28VIN, 70W in meno di 4,5cm2
- Soluzione completa che richiede un numero ridotto di condensatori ceramici e resistenze
- Package BGA di 15 x 15 x 5,01mm con dissipatore di calore integrato
- Corrente di uscita completa (5A, 5A, 4A) su tutti i canali con temperatura ambiente di 65°C e flusso d'aria di 200 LFM, nessun dissipatore esterno
- Funzionamento VIN nel range da 4,75V a 28V con ingressi separati per ciascun canale
- VOUT con precisione di ±1,5% regolabile dall'utente tra:
  Canale 1: da 0,8V a 5,5V
  Canale 2: da 0,8V a 5,5V
  Canale 3: da 0,8V a 13,5V
  I canali 1 e 2 possono essere collegati in parallelo per fornire fino a 10A

SUGGERITI DA IE cloud

Pillole di microcontrollori PIC

Per imparare a programmare i PIC utilizzando il linguaggio C.

ACQUISTALO ORA!

Secure Payments By PayPal

ARDUINO Projects

190 pagine di progetti con Arduino in pdf con spiegazioni dettagliate e codici sorgente.

ACQUISTALO ORA!

Secure Payments By PayPal